A STOCHASTIC EM ALGORITHM FOR G-RHO FAMILY ACCELERATED FAILURE TIME MODEL WITH RANDOM EFFECTS

by

KyungAh Im

BS, Carnegie Mellon University, 1996

MS, University of Pittsburgh, 2002

Submitted to the Graduate Faculty of
the Department of Biostatistics
Graduate School of Public Health
in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy

University of Pittsburgh

2013

UNIVERSITY OF PITTSBURGH

GRADUATE SCHOOL OF PUBLIC HEALTH

This dissertation was presented

by

KyungAh Im

It was defended on

December 7, 2012

and approved by

Jong-Hyeon Jeong, Ph.D
Professor
Department of Biostatistics
Graduate School of Public Health
University of Pittsburgh

Abdus Wahed, Ph.D
Associate Professor
Department of Biostatistics
Graduate School of Public Health
University of Pittsburgh

Howard E. Rockette, Ph.D
Professor Emeritus
Department of Biostatistics
Graduate School of Public Health
University of Pittsburgh

Steven H. Belle, Ph.D., M.Sc.Hyg.
Professor
Department of Epidemiology
Graduate School of Public Health
University of Pittsburgh

Dissertation Director: Jong-Hyeon Jeong, Ph.D
Professor
Department of Biostatistics
Graduate School of Public Health
University of Pittsburgh

A STOCHASTIC EM ALGORITHM FOR G-RHO FAMILY ACCELERATED FAILURE TIME MODEL WITH RANDOM EFFECTS

KyungAh Im, PhD

University of Pittsburgh, 2013

We propose an accelerated failure time model with random effects for correlated or clustered survival time data. We assume that the error distribution belongs to the family of G^ρ distribution and random effects follow multivariate normal distribution. The G^ρ family distribution allows us to model "attenuating" or "converging" hazard functions over time, which represent a type of non-proportional hazards, with the special case of $\rho = 1$ corresponding to the proportional odds model. In G^ρ family distribution, the larger the value of ρ the higher the degree of non-proportionality in the data. Thus, estimating ρ as a regression parameter instead of assuming a priori fixed value allows us more flexibility handling many different types of non-proportional converging hazards occurring in practice. We utilize EM algorithm for estimation. More specifically, Stochastic Expectation-Maximization(StEM) algorithm is used to maximize the complete data log-likelihood in the presence of random effects. The conditional expectation in the classical E-step is replaced by a stochastic draw from the posterior distribution of the latent variable via Gibbs sampler method. The computational complexity of the likelihood then can be avoided by maximizing the pseudo-complete data. We also examine the robustness of the estimated fixed effects and the estimated variance components when the error distribution or distribution of the random effects is misspecified through simulation studies, followed by an application to a clinical trial dataset.

Public health significance: The proposed method enables researchers 1) to model dependent or clustered survival data which arise frequently in medical research; for example, in familial studies or multi-center clinical trials. 2) to model attenuating hazards which is a type of non-proportional hazard. 3) to reduce bias in estimating treatment effects in the presence of non proportional hazards and unobserved heterogeneity. The proposed method contributes to more accurate estimation of important covariate effects (such as treatment effects) in practical settings such as in randomized clinical trial.

KEYWORDS: Accelerated failure time model, G^ρ family distribution, frailty, random effects, Stochastic Expectation-Maximization(StEM), ARMS, Gibbs sampler.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1.0 INTRODUCTION

The field of survival analysis is very rich and has been growing tremendously in the 20th century and still very active field in statistical community. Fleming and Lin (2000) review the past developments and future directions in survival analysis in clinical trials. These two authors mentioned that Kaplan-Meier method [35], the log-rank statistic [51], the Cox proportional hazards model [10] and the counting-process martingale theory [1] provided the most profound impact on clinical trials. In addition, much progress has been made and further developments are expected in many other areas: including the accelerated failure time model, multivariate failure time data, interval-censored data, dependent censoring, dynamic treatment regimes and causal inference, joint modeling of failure time and longitudinal data, and Bayesian methods [19, 56].

Survival data concern measuring time to a particular event of interest. Especially in clinical trials or any observational cohort studies in epidemiology, one of the most prevalent study endpoint is a survival outcome such as time to death, time to major organ failure, time to serious infection, heart attack, stroke, time to occurrence of disease or complication or symptom.

A special feature of survival data is "censoring" which means information is incomplete in the sense that we may not observe the true time to event for some subjects during the course of the study and this needs to be taken into account for analysis. Another feature is "conditioning". For a simple example given by Hougaard (1999), calculating a probability for a person to die at age 75 does not make sense

if that person had died at age 70. What is relevant is the truncated distribution of the lifetime after age 75 years, i.e., the distribution given that the lifetime exceeds 75 years. In this regard among other things, the concept of "hazard" became very appealing and important in this field. The hazard function is the probability of death (or event) within a *short* interval, given that the person was alive (or had not experienced the event) at the beginning of the interval, usually denoted by $\lambda(t)$ or $h(t)$:

$$h(t) = \lim_{\Delta t \to 0} \frac{Pr(t < T < t + \Delta t \,|\, T \geq t)}{\Delta t}, \tag{1.1}$$

which is basically a limiting conditional probability as the short interval becomes extremely small. The hazard function is also called the instantaneous death rate, the intensity rate, or the force of mortality. It is a rate since it is a function of time. Of note, the hazard rate is not a probability per se because it can exceed 1. Without conditioning on $T \geq t$ in (1.1), this quantity becomes the probability density function $f(t)$. Another quantity of interest is the survivor function, which is the probability that the survival time T is greater than t, and

$$S(t) = Pr(T > t) = 1 - F(t)\,, \tag{1.2}$$

where $F(t) = Pr(T \leq t)$ and that we have following basic mathematical relationships

$$h(t) = \frac{f(t)}{S(t)} = -\frac{d}{dt} \log S(t)\,, \tag{1.3}$$

The cumulative hazard can be obtained by integrating both sides in equation (1.3) for the continuous T in the interval $(0, t)$

$$H(t) = -\log S(t)\,, \tag{1.4}$$

and the density function is

$$f(t) = -\frac{d}{dt} S(t). \tag{1.5}$$

1.1 BASIC REGRESSION MODELS IN SURVIVAL ANALYSIS

One of the interests in survival analysis is to relate a set of explanatory variables to the survival outcome. A popular model is the semi-parametric Cox proportional hazards model [10] for which the hazard for a j-th individual is denoted by

$$h_j(t) = h_0(t) \exp(\beta^T Z_j), \tag{1.6}$$

which is a product of a baseline hazard $h_0(t)$ and a term $\exp(\beta^T Z_j)$ that depends on the observable covariates Z_j. In this model, the ratio of the hazards for two individuals is constant over time. In other words, the hazard for any individual is a fixed proportion of the hazard for any other individual. Thus, it is a proportional hazards model. For example, the difference in the risk of experiencing an event (i.e., hazard function) between the two individuals (lines) at any given time point stays constant regardless of the shape of the hazard functions as in Figure 1.1 (a) through (c).

The semi-parametric Cox proportional hazards model has become popular mostly because the partial likelihood estimation of the model does not require a parametric form for the baseline hazard function $h_0(t)$ to estimate the effect of parameters β. It describes relative risks which are independent of time but depend on the values of covariates. Cox models have been extensively used in medical research.

However, there may be factors other than the observed covariates that significantly affect the distribution of survival time or pertinent covariates are omitted in the model due to lack of knowledge. This is commonly known as *unobserved random heterogeneity* in the survival analysis literature. The first model of mortality data to include of this kind of individual heterogeneity was introduced by Vaupel et al (1979). The model allowed each individual to have a frailty term to account for individual differences in mortality hazard rate. The individual frailty was assumed to be a random variable and

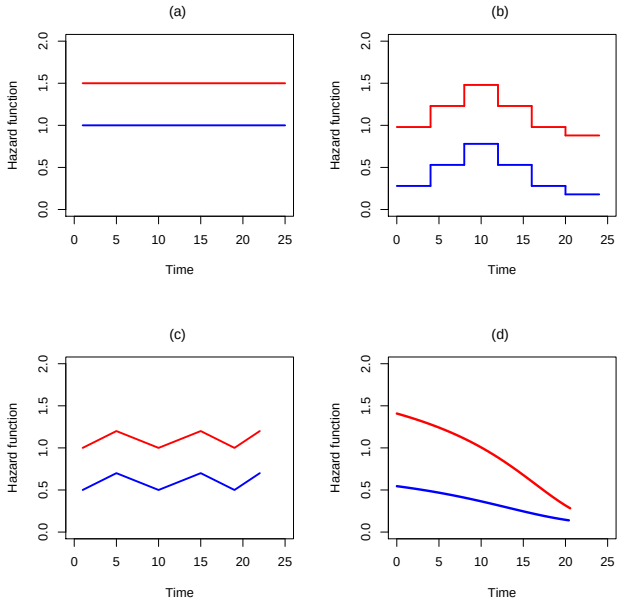

Figure 1.1: Proportional and non-proportional hazards

defined as a positive constant, say U_j, which imposed a multiplicative effect on the force of mortality (hazard) so that the hazard for a j-th individual was defined as

$$h_j(t) = U_j h_0(t). \tag{1.7}$$

For example, an individual with a frailty of $U_j = 2$ is twice as likely to die at any particular age and time, as the "standard" individual whose hazard is denoted by $h_0(t)$. Frail individuals with high values of U_j tend to die first. Thus, Vaupel et al. coined the term "frailty" for the random variable U_j which was assumed to follow a gamma distribution for a mathematical convenience. This model (1.7) is regarded as a univariate frailty model in survival analysis literature. In his study, individuals were independent of each other and had each frailty term. The main objective of Vaupel et al. was to show that population mortality hazard rates do not reflect the the mortality hazard rates of individuals from that population. Differences between population hazard and individual hazards become larger

as variance (heterogeneity) in the population becomes greater. The important message of Vaupel's work is that heterogeneity in the hazard function of different individuals should not be neglected. The assumptions that the frailty is independent of age (which means frailty is not dependent on time) and that it acts multiplicatively on an underlying hazard have been taken as the basis for much subsequent work on random heterogeneity in survival analysis including clustered survival data [36].

The most common frailty model is the shared frailty model which is an extension of the Cox proportional hazards regression model. In this model, all individuals within a group or cluster share a common unobservable random effect, the frailty, which acts multiplicatively on each individual's hazard rate, i.e., for the jth individual in a cluster i,

$$h_{ij}(t) = U_i h_0(t) \exp(\beta^T Z_{ij}), \tag{1.8}$$

where U_i is the frailty and it is usually an exponentiated function of random effects (such as $\exp(b_i)$) to be a positive quantity ($U_i > 0$). U_i is generally assumed to be an independent and identically distributed sample from a distribution with known mean and some unknown variance θ. In equation (1.8), the dependence on the covariates is further parameterized in the Cox model setting under the standard assumption of the hazard being proportional over time. The baseline hazard function in this model can be specified (such as Weibull or piecewise constant) or arbitrary which yields parametric or semi-parametric Cox frailty models, respectively. In univariate frailty models, one often assumes that, conditional on the unobserved frailty, individuals within a cluster are independent of each other. This assumption can be relaxed when we consider the multivariate frailty model setting.

A proportional hazards model with frailty has been considered by many authors in the literature and different frailty distributions have been considered including the gamma distribution by Klein (1992) [37] and Clayton (1991) [9], the positive-stable distribution and the inverse-Gaussian distri-

bution by Hougaard (1986a, 1986b) [27, 28], the log-normal distribution by McGilchrist and Aisbett (1991) [52], and the normal distribution by Vaida and Xu (2000) [69]. Of note, the frailty model in equation (1.8) can lead to hazards not being proportional except when frailty follows a positive stable distribution (Hougaard 1995, 1999) [29, 30].

Ignoring frailty can lead to non-proportional hazards in the proportional hazards parameterization and can yield biased regression coefficient estimates $\hat{\beta}$ [63]. In general, "frailty model" in survival analysis literature refers to the random effects model in the Cox model setting and is used to account for overdispersion or correlation in the survival data.

Rather than having the hazard function be described as a function of explanatory variables, it is possible to let the explanatory variables act directly on the survival time via a scale factor under the accelerated failure time model setting [30]. There are two basic ways to express this type of model. An accelerated failure time representation and a log-linear model representation. The general form of the log-linear model representation is presented with more detail in the next chapter. For comparison with the Cox model we will use an accelerated failure time model representation using hazard functions to describe a general accelerated failure time model in this section, that is, for j-th individual

$$h_j(t) = h_0(t \times \exp(\eta^T Z_j)) \exp(\eta^T Z_j), \tag{1.9}$$

where η is the vector of fixed effects regression coefficients including an intercept term and Z_j is the vector of observed covariates. The factor $\psi_j = \exp(\eta^T Z_j)$ is also called an acceleration factor indicating how a change in covariate values changes the time scale from the baseline time scale. This implies that the models can be interpreted in terms of the speed of progression of a disease. The general accelerated failure time model that incorporates a frailty term U_i can be written

$$h_{ij}(t) = U_i h_0(t \times \exp(\eta^T Z_{ij}) U_i) \exp(\eta^T Z_{ij}), \tag{1.10}$$

where $U_i = \exp(b_i^T W_{ij})$. Here, b_i represents a random effect assigned to a cluster i. These random effects are unobserved (latent or missing data) and are estimable quantities. W_{ij} represents either a design matrix or a subset of the covariate data Z, depending on the structure of frailty or random effects in the model. If we let $\psi^\star$ be a function of covariates Z without an intercept term and ψ be a function of the covariates Z with an intercept term then we have the following four basic types of regression models in survival analysis (Table 1).

Table 1: Basic regression models in survival analysis

Cox model	Accelerated Failure Time model
$h_j(t) = h_0(t)\psi^\star$	$h_j(t) = h_0(t \times \psi)\psi$
Cox model with Frailty	Accelerated Failure Time model with Frailty
$h_{ij}(t) = U_i h_0(t)\psi^\star$	$h_{ij}(t) = U_i h_0(t \times \psi U_i)\psi$

Although the parametric proportional hazards models are widely used in the analysis of survival data, the accelerated failure time models are an important alternative in circumstances for which the proportional hazards assumption is not tenable. With a wider range of survival time distributions the accelerated failure time model can be used to accommodate various departures from the proportional hazards assumption including crossing hazards and attenuating hazard ratios such as in Figure 1.1 (d).

When the deviations from proportional hazards are due to unaccounted random heterogeneity (e.g., omitted important covariates in the model) the accelerated failure time model parameters are more robust than the Cox proportional hazards model parameters, and Hougaard (1999) noted that this is a major drawback of Cox proportional hazards model [30]. In addition, regression parameters in the proportional hazards model are more sensitive to the distribution of the frailty. Keiding et al (1997) [36] show that the regression parameters of the AFT model are robust against the misspecification

of the frailty distribution. This finding is further supported by the empirical results from simulation studies of Lambert et al (2004) [44].

For clustered data, an AFT model with random effects can be considered as a classical linear mixed effects model of Laird and Ware (1982) [43] with the logarithmic link function. The practical interpretation under this model is more straightforward than the Cox model with random effects because one does not have to resort to the hazard function. In addition, the fixed effects regression coefficient can have a population-averaged (marginal) interpretation of a given covariate as well as the a cluster specific (conditional) interpretation given the random effect parameter. Also, a simple one-way random effects AFT model (such as a random intercept model) can have the natural decomposition of overall variation of the response into within-individual (cluster) and between-individual (cluster) variations.

Anderson and Louis (1995) [3] show an example using an AFT model under a scale change random effects model for bivariate survival data using the Gompertz distribution for baseline survival distribution and gamma frailty distribution. Klein et al (1999) [38] utilize an AFT model for the bivariate survival time to occurrences of coronary heart disease among sibling groups selected from the Framingham Heart Study. These authors use the log-normal distribution for both the baseline survival distribution and the frailty distribution. More recently, Lambert et al (2004) [44] explored the parametric random effects AFT models for kidney transplant data including the gamma, inverse-Gaussian, log-normal, log-logistic and Weibull distributions for the baseline survival distribution and developed a parametric mixture distribution for baseline survival distribution. Komarek et al (2007) [41] explored random effects AFT model with a normal mixture error distribution in Bayesian framework. However, the G^{ρ} family of distribution, which can handle non proportional attenuating hazard functions, has not been considered in the literature for the baseline survival distribution in the accelerated failure

time model setting. In this dissertation, we propose to model the distribution of the error term using the family of G^ρ distributions for the clustered right-censored survival data and to develop an estimation procedure based on Markov Chain Monte Carlo method. The organization of this dissertation is as follows. We present some basic details about a log-linear model representation of the AFT model, the family of G^ρ distributions and other distributional assumptions in chapter 2. Chapter 3 presents the estimation and inference algorithm. We investigate finite sample performance of the estimator through simulations in chapters 4. We will also examine, through simulation studies, the robustness of the estimated fixed effects and the estimated variance components when the error distribution or distribution of the random effects is misspecified. Chapter 5 presents the application of our model to a randomized clinical trial followed by discussion in chapter 6.

2.0 ACCELERATED FAILURE TIME MODEL

Let $S_0(t)$ be a survivor function for the placebo group (Z=0) and $S_1(t)$ be a survivor function for the treatment group (Z=1). In general, an AFT model is described using an accelerated failure time representation using the survival function at a time t:

$$S_1(t) = S_0(t * \psi), \tag{2.1}$$

where $\psi = \exp(\theta^T Z)$ is an acceleration (or deceleration) factor that determines how much to move on the time scale. ψ is a function of covariates Z and a vector of regression coefficients $\theta^T = (\theta_1, \ldots, \theta_p)$. This implies that the shape of survivor function stays the same but shifts to the left or to the right on the time scale. Corresponding hazard and density functions are $h_1(t) = \psi h_0(t*\psi)$ and $f_1(t) = \psi f_0(t*\psi)$, respectively.

For example, the AFT model states that the survival function of an individual in the treatment group $S_1(t)$ with covariate $Z = 1$ at a time t is the same, by shifting the time scale, as the survival function of an individual in the placebo group $S_0(t)$ with a baseline survival function at a time $t \times \exp(\theta^T Z)$. That is,

$$S_1(t) = S_0(t \times \exp(\theta^T Z)). \tag{2.2}$$

If the factor $\exp(\theta^T Z)$ is greater than 1 in the equation (2.2), the event-free time (survival time) for the treatment group is shorter than that of the placebo group. Treatment (covariate) speeds up the process (here, the process means the expected time to failure or event). Thus, it's called an acceleration factor

(Figure 2.1.(a)). If the factor $\exp(\theta^T Z)$ is less than than 1 in the equation (2.2), the event-free time for the treatment would be longer than that of placebo group, i.e., treatment slows down the process. Thus, it's called a deceleration factor (Figure 2.1.(b)). For clinical trial settings in which the outcome is death we want the treatment to slow down the process or lengthen the event-free time, so we need a deceleration factor; whereas, in a clinical trial where the end-point outcome is time to recovery we want the treatment to reduce time to outcome. so we hope for an acceleration factor.

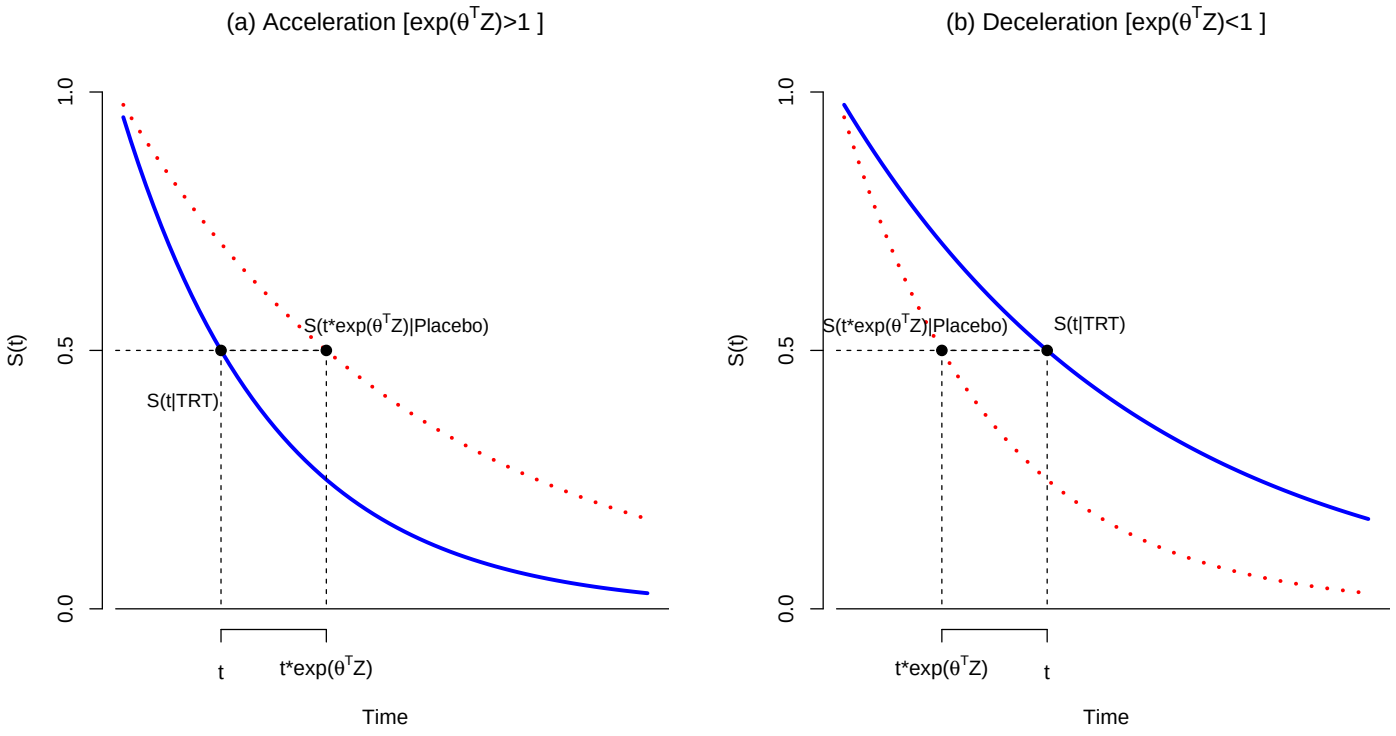

Figure 2.1: Acceleration and Deceleration in the AFT model

The acceleration (or deceleration) factor can also be interpreted in terms of the median survival times of subjects on the treatment and placebo groups. Let $t_1(50)$ and $t_0(50)$ be such two median survival times, respectively. These values are such that $S_1\{t_1(50)\} = S_0\{t_0(50)\} = 0.5$. Under an AFT

model, we have $S_1\{t_1(50)\} = S_0\{t_1(50) * \psi\}$ where ψ can be any positive quantity. Thus, it follows that $t_1(50) = t_0(50)/\psi$. The median survival time of a subject on the treatment is $1/\psi$ times that of a subject on the placebo. The same argument can be used for any percentile of the survival time distribution.

2.1 LOG-LINEAR FORM OF THE AFT MODEL

Consider two random variables T_1 and T_0 for survival times of treatment and placebo group, respectively. Under the accelerated failure time model setting in 2.1, an individual having survival time t under the treament group $(Z = 1)$ would have survival time $t\boldsymbol{\zeta}$ under the placebo group $(Z = 0)$, i.e., the corresponding random variables are related by $T_1 = T_0\boldsymbol{\zeta}$ where ζ can be some positive quantity. Now suppose further the constant $\boldsymbol{\zeta}$ is a function of some observed covariates Z: $\boldsymbol{\zeta}(Z)$. Then we have $T_1 = T_0\boldsymbol{\zeta}(Z)$. Taking a natural logarithm,

$$\log(T) = \log T_0 + \log \boldsymbol{\zeta}(Z) + \boldsymbol{\epsilon}, \tag{2.3}$$

where $\boldsymbol{\epsilon}$ is error term since T is a random variable. It is usually assumed that the expectation of the error distribution is zero $E(\boldsymbol{\epsilon}) = 0$ and independent of observed covariates Z. Since we want $\boldsymbol{\zeta}(Z) \geq 0$ and $\boldsymbol{\zeta}(Z = 0) = 1$ a natural candidate for $\boldsymbol{\zeta}(.)$ is $\exp(\beta^T Z)$. Then an AFT model can be expressed in the log-linear model representation:

$$\log(T) = \mu + \beta^T Z + \tau\boldsymbol{\epsilon}, \tag{2.4}$$

where μ is an intercept term which is the baseline log survival time ($\log T_0$) and τ is a scale parameter.

We can also obtain an acceleration factor using the regression coefficients β from the log-linear AFT model (2.3). Consider the survivor function of an individual j,

$$\begin{aligned} S_j(t) &= Pr(T_j \geq t) \\ &= Pr\left(\exp\{\mu + \beta^T Z_j + \tau\epsilon_j\} \geq t\right) \\ &= Pr\left(\exp\{\mu + \tau\epsilon_j\} \geq t/e^{\beta^T Z_j}\right) \\ &= S_0(t\, e^{-\beta^T Z_j})\,, \end{aligned} \tag{2.5}$$

which is the general form of the survivor function for the jth individual in an accelerated failure time model, if $\exp\{\mu + \tau\epsilon_j\}$ follows the $S_0(.)$ distribution. Here, the acceleration factor is the term $\exp(-\beta^T Z_j)$. Comparing to the factor from the accelerated failure time representation which is $\exp(\theta^T Z)$ we have the following relationship,

$$\theta = -\beta\,. \tag{2.6}$$

This indicates that one needs to reverse the sign of the regression coefficients from the log-linear AFT model to calculate an acceleration (or deceleration) factor.

The AFT model can be represented as a location-scale model using $\epsilon_j = \dfrac{\log t - (\mu + \beta^T Z_j)}{\tau}$ so the survivor function of $\log T$ can be found by using the distribution of the error ϵ_j:

$$S(t) = S_0(\frac{\log t - (\mu + \beta^T Z_j)}{\tau}) \tag{2.7}$$

where $\epsilon_j \sim S_0(\epsilon)$.

Commonly used distributions of ϵ are normal $S_0(\epsilon) = 1 - \Phi(\epsilon)$, extreme value $S_0(\epsilon) = \exp(-e^{\epsilon})$, and logistic $S_0(\epsilon) = (1 + e^{\epsilon})^{-1}$, which correspond to log-normal, Weibull and log-logistic distributions for survival time T.

2.2 AFT MODEL WITH RANDOM EFFECTS

Using the log-linear model notation from equation (2.4), we can incorporate random effects by adding the term $b_i^T W_{ij}$,

$$\log T_{ij} = \beta^T Z_{ij} + b_i^T W_{ij} + \tau * \epsilon_{ij}, \tag{2.8}$$

where i denotes a group membership and j denotes an observation, T_{ij} is the failure time, Z_{ij} and W_{ij} are covariate vectors, β is the vector of fixed effects, and b_i is the vector of random effects for the ith group. The intercept μ is absorbed in the β vector and the scale parameter τ is assumed to be one. The random effects AFT model can be considered as a classical linear mixed model of Laird and Ware (1982) [43] with the logarithmic link function. The log-linear formulation of the model above can also be used to give a general form of the survivor function. With the parametric distribution of the error terms we can express the survivor function of the jth individual in the ith group in terms of the error random variable ϵ_{ij},

$$\begin{aligned} S_{ij}(t) &= Pr(T_{ij} \geq t_{ij}) \\ &= Pr(\log T_{ij} \geq \log t_{ij}) \\ &= Pr(\beta^T Z_{ij} + b_i^T W_{ij} + \epsilon_{ij} \geq \log t_{ij}) \\ &= Pr(\epsilon_{ij} \geq \log t - \beta^T Z_{ij} - b_i^T W_{ij}) \\ &= S_{\epsilon_{ij}}(\log t - \beta^T Z_{ij} - b_i^T W_{ij}). \end{aligned} \tag{2.9}$$

The cumulative hazard function of the distribution of T_j is given by,

$$\begin{aligned} H_{ij}(t) &= -\log S_{ij}(t) \qquad (2.10) \\ &= -\log S_{\epsilon_{ij}}(\log t_{ij} - \beta^T Z_{ij} - b_i^T W_{ij}) \\ &= H_{\epsilon_{ij}}(\log t_{ij} - \beta^T Z_{ij} - b_i^T W_{ij}). \qquad (2.11) \end{aligned}$$

By differentiating $H_{ij}(t)$ with respect to t the hazard function of ϵ is

$$h_j(t) = \frac{1}{t} h_{\epsilon_j}(\log t - \beta^T Z - b^T W). \tag{2.12}$$

2.3 G^ρ BASELINE SURVIVAL DISTRIBUTION

We study a random effects AFT model for right-censored clustered data for which the error distribution follows the family of G^ρ distributions (Harrington and Fleming, 1982) [26]. The family of G^ρ distributions includes the logistic distribution as a special case and gives rise to the generalized odds-rate model studied by Dabrowska and Doksum (1988) and Jeong et al (2003) [11, 34] in the Cox model setting. The survivorship functions of the family of G^ρ distributions are given as follows:

$$S_0(t) = \exp(-e^t) \quad (\rho = 0), \tag{2.13}$$

$$S_\rho(t) = (1 + \rho e^t)^{-1/\rho} \quad (\rho > 0, \ -\infty < t < \infty). \tag{2.14}$$

When $\rho = 0$ the survivor function is the extreme value distribution which leads to a proportional hazards model. When $\rho > 0$ the hazard functions for different groups converge as $t \to \infty$. A closer look at these two expressions reveals that the family of G^ρ distributions consists of an extreme value distribution and a class of logistic distributions combined through the parameter ρ. Harrington and Fleming (1982) pointed out that the family of G^ρ distributions is an important subset of the generalized-F family of distributions. The family of G^ρ distributions is useful in modeling attenuating (or converging) hazard functions that often arise in practice (see figure 2.2), and the special case of $\rho = 1$ corresponds

to the proportional odds model. We re-parameterize $\rho = \exp(\alpha)$ and use $\alpha = \log \rho$ in the simulation studies and modeling. Only $\rho > 0$ $(-\infty < \alpha < \infty)$ is considered throughout the dissertation. $\alpha = \log \rho$ is estimated from the data. Then, the survivor function of the random variable ϵ_{ij} for the jth individual in group i from equation (2.9) and (2.14) can be written

$$\begin{aligned} S_{\epsilon_{ij}}(\epsilon) &= \left(1 + e^{\alpha+\epsilon_{ij}}\right)^{-e^{-\alpha}} \\ &= \left[1 + e^{\alpha+(\log t - \beta^T Z_{ij} - b_i^T W_{ij})}\right]^{-e^{-\alpha}} \\ &= S_{ij}(t). \end{aligned} \tag{2.15}$$

And hazard function of the random variable ϵ_{ij} is

$$\begin{aligned} h_{\epsilon_{ij}}(\epsilon) &= -\frac{\partial}{\partial \epsilon} \ln S_{\epsilon_{ij}}(\epsilon) \\ &= -\frac{\partial}{\partial \epsilon} \left[(-e^{-\alpha}) \ln\left(1 + e^{\alpha+\epsilon_{ij}}\right)\right] \\ &= e^{-\alpha} \frac{1}{(1 + e^{\alpha+\epsilon_{ij}})} e^{\alpha+\epsilon_{ij}} \\ &= \frac{e^{\epsilon_{ij}}}{1 + e^{\alpha+\epsilon_{ij}}} \\ &= \frac{e^{\log t - \beta^T Z_{ij} - b_i^T W_{ij}}}{1 + e^{\alpha+\log t - \beta^T Z_{ij} - b_i^T W_{ij}}}. \end{aligned} \tag{2.16}$$

Then the hazard function of t for an individual j using $h_{\epsilon_j}(\epsilon)$ can be written from equation (2.12)

$$\begin{aligned} h_{ij}(t) &= \frac{1}{t} h_{\epsilon_{ij}}(\epsilon_j) \\ &= \frac{1}{t} \left(\frac{e^{(\log t - \beta^T Z_{ij} - b_i^T W_{ij})}}{1 + e^{\alpha+(\log t - \beta^T Z_{ij} - b_i^T W_{ij})}} \right). \end{aligned} \tag{2.17}$$

The density function is then given by

$$\begin{aligned} f_{ij}(t) &= -dS_{ij}(t)/dt \\ &= h_{ij}(t)S_{ij}(t) \\ &= \frac{1}{t}\left(\frac{e^{(\log t-\beta^T Z_{ij}-b_i^T W_{ij})}}{1+e^{\alpha+(\log t-\beta^T Z_{ij}-b_i^T W_{ij})}}\right)\times\left[1+e^{\alpha+(\log t-\beta^T Z_{ij}-b_i^T W_{ij})}\right]^{-e^{-\alpha}}. \end{aligned} \tag{2.18}$$

Figure 2.2 shows different scenarios of converging hazard functions as the value of α varies $\alpha = (1.0, 0.5, 0.0, -0.69)$ for the Figures (a),(b),(c) and (d), respectively. $\alpha = 0.0$ corresponds to a proportional odds rate model. These plots are based on the simulated data from a fixed effects AFT model $Y = \mu + \beta_1 X_1 - \beta_2 X_2 + \epsilon$ where X_1 is a continuous variable representing some baseline measurement such as blood pressure and X_2 is a binary variable indicating treatment group. The dotted line represents the hazard function of a subject in the treatment group A and the solid line represents the hazard function of a subject in the treatment group B.

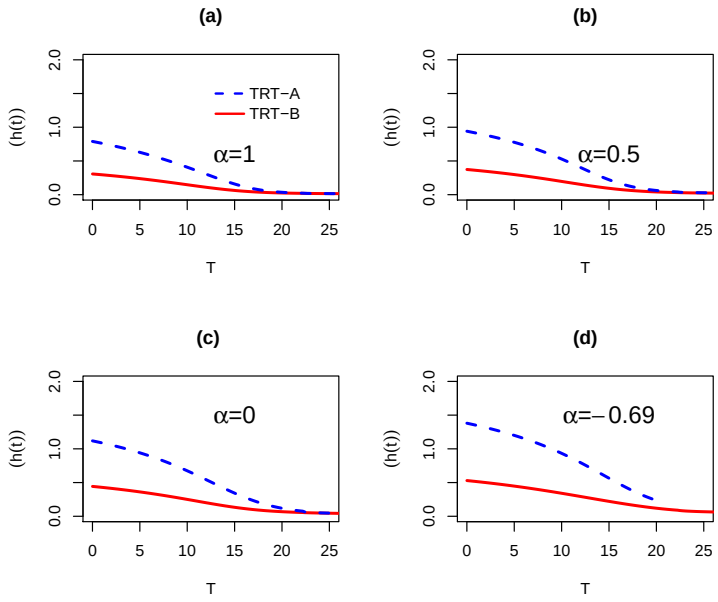

Figure 2.2: Attenuating hazard plots from an AFT model with G^ρ error distribution.

2.4 RANDOM EFFECTS STRUCTURE AND DISTRIBUTION

Depending on the structure of frailty we obtain different types of random effects model.

Shared frailty model in univariate setting: A random effect variable b_i is assigned to each cluster i, so all the members within a cluster i have the same random effect. This causes correlation among the members of a cluster. Random effects variables for all of the clusters $b = (b_1, \ldots, b_G)$ (where G is the total number of clusters) follow a common distribution. However, by conditioning on the random effects it is assumed that observations are independent. One typical example of the application of this model would be modeling the heterogeneity across clusters such as study sites. Another example would be repeated measurement data within each individual study subject. By assigning a random effect for each subject the correlation among repeated measurement data can be taken into account in the analysis. ***Shared frailty model in multivariate setting***: We can also assign more than one random variable within a cluster so that some members of the cluster i can be assigned to one random variable b_{1i} whereas the rest of the members are assigned to another random variable b_{2i}. These two random variables b_{1i} and b_{2i} can be independent or correlated. This is an example of a bivariate Shared Frailty model. An example would be family data. One random variable can be assigned for parents and another for the children so that they would no longer be constrained to have a common frailty. Since they all belong to the same cluster (family) we can imagine these two random effects could be correlated. (Xue and Brookmeyer 1996 [73]).

Consider a random effects AFT model in log-linear model representation: $\log T_{ij} = \beta^T Z_{ij} + b_i^T W_{ij} + \epsilon_{ij}$ where b_i is the random effect from the ith cluster, and Z_{ij}, W_{ij} are the covariate vectors for the fixed and random effects. For a simple univariate shared frailty model, W_{ij} is the vector of '1's which represents the cluster effect on the baseline log survival time, that is, $\log T_{ij} = \beta_0 + \beta_1 Z_{1ij} + b_{0i} + \epsilon_{ij}$ for

a covariate Z_{1ij}. This is equivalent to a random intercept model. For a bivariate shared frailty model, for example, $\log T_{ij} = \beta_0 + \beta_1 Z_{1ij} + b_{1i} W_{1ij} + b_{2i} W_{2ij} + \epsilon_{ij}$ where $W_{ij} = [W_{1ij}, W_{2ij}]^T$ are the indicator variables for two sub-clusters membership within a cluster i. It fits a random intercept $\beta_0 + b_{1i}$ for sub-cluster $1i$ and another random intercept $\beta_0 + b_{2i}$ for sub-cluster $2i$ within a cluster i.

Mixed effects model setting: Another type of random effects model can be chosen in the context of a classical linear mixed effects model by Laird and Ware (1982) [43]. That is, the matrix W is a subset of covariate vector Z. For a simple example, let $W = [1, Z_{1ij}]^T$, $\mathbf{Z} = [1, Z_{1ij}]^T$ and $b = [b_{0i}, b_{1i}]$, resulting in a mixed effects model $\log T_{ij} = \beta_0 + \beta_1 Z_{1ij} + b_{0i} + b_{1i} Z_{1ij} + \epsilon_{ij}$. In this model, b_{0i} is the main cluster effect and b_{1i} represents the interaction between the cluster and the respective covariate Z_{1ij}. The coefficient β_1 represents the main effect of the covariate Z_{1ij}. If we consider Z_{1ij} to be, for example, the treatment indicator in a multi-center clinical trial then b_{1i} represents an interaction between treatment and cluster, thus it is modeling the treatment heterogeneity across clusters or institutional treatment effects. This is one example of a mixed effects model. The matrices Z and W can also be mutually exclusive.

Thus, a log-linear mixed effects AFT model as in the equation (2.8) gives more options for modeling the random effects. It can accommodate a simple univariate shared frailty model, multivariate shared frailty model, the nested frailty structure, and true mixed effects models by allowing other exploratory variables of interests to be random effects.

In this dissertation we will assume that the random effects $b_i, (i = 1, \ldots, G$ cluster$)$ follow multivariate normal distribution with known mean $E[b_i] = 0$ and unknown covariance matrix Σ. The number of random effects within a cluster can be up to d (where $d \leq n_i$ the number of members in a cluster).

$$b_i \sim N(0_{d\times 1}, \Sigma_{d\times d})\,, \tag{2.19}$$

where $\Sigma = \begin{pmatrix} \sigma_1^2 & \cdots & \rho\sigma_d\sigma_1 \\ \vdots & \ddots & \vdots \\ \rho\sigma_1\sigma_d & \cdots & \sigma_d^2 \end{pmatrix}$. Initially we assume a diagonal matrix for variance components of the random effects and then try to estimate a covariance component in the bivariate random effects model setting in simulation study. The distribution of random effects can be other than the normal distribution, such as the multivariate t-distribution.

3.0 ESTIMATION AND INFERENCE

3.1 BACKGROUND

The regression analysis of independent observations from survival data with a log-linear model has been studied extensively. The two classical approaches to semi-parametric AFT models for uncorrelated data are the Buckley-James method (1979) [5] and the method based on estimating equation using linear rank statistics (Tsiatis 1990) [67]. Wei (1992) [72] reviews these two methods. More recently, Komarek, Lesaffre and Hilton (2005) [40] proposed an AFT model with the error distribution estimated by a penalized maximum likelihood method.

For correlated failure time data, Pettitt (1986) [59] considered a normal mixed effects model approach to estimate the random subject effects (b_i) for the logarithm of right-censored repeated measures data from a matched skin graft study and used maximum likelihood estimation via the Expectation-Maximization (EM) algorithm of Dempster, Laird and Rubin (1977) [12]. This is an example of the one-way random effects model or the random intercept only model. Anderson and Louis (1995) [3] considered maximum likelihood estimation of a scale change random effects model using numerical integration to evaluate the likelihood function.

Hughes (1999) [32] adapted a Monte-Carlo EM (MCEM) algorithm based on Gibbs sampling for the linear mixed effects model of Laird and Ware (1982) [43] and provided a general solution to

accommodate both right- and left-censored data. Hughes (1999) extended Pettitt's model (1986) in a repeated measures study in which an individual is a unit of a cluster in the area of AIDS research. Both Pettitt (1986) and Huges (1999) considered Gaussian errors and random effects in their models. Klein, Pelz, and Zhang (1999) [38] also considered the random effects AFT model with Gaussian error distribution using the Newton-Raphson method. Ha et al (2002) [25] applied the hierarchical-likelihood approach of Lee and Nelder (1996) [48] for right censored survival data in a normal linear mixed models setting (again, a random intercept model: $\log T_{ij} = \beta^T Z_{ij} + b_i + \epsilon_{ij}$).

Pan and Louis (2000) [58] proposed a different approach for the same linear mixed effects model of Hughes (1999) where only the random effects were treated as missing and incorporated the Buckley-James methods (1979) [5] to handle the censored data. Their procedure iterates between (a) estimating the marginal distribution of $\log T_{ij} - \beta^T Z_{ij}$ using Kaplan-Meier estimation and imputing censored event times, and (b) estimating regression coefficients using a Monte Carlo EM algorithm. Pan and Louis (2000) [58] only considered a univariate random effects model in their approach. Although Pan and Louis (2000) [58] assumed normality (both error and random effects follow Gaussian distribution) their method is considered to be semi-parametric due to the incorporation of nonparametric least-squares estimation and the Kaplan-Meier estimator.

Early estimation approaches for the semi-parametric AFT model with correlated survival data are based on the estimating equation. Lee et al. (1993) [47] considers the log-linear model for the jth observation in the ith individual: $\log T_{ij} = \beta^T Z_{ij} + \epsilon_{ij}$ and first developed a marginal approach to correlated failure time data by using GEE method of Liang and Zeger (1986) [49]. Their main interest was to obtain the average response for the population, thus the correlation in the data is treated as a nuisance parameter. Pan and Connett (2001) [57] adopted the data augmentation method of Tanner and Wong (1987a) [66] to impute censored failure times and fit either a marginal model

based on the GEE approach with the sandwich estimator for covariance or a linear mixed effects model with restricted maximum likelihood estimator (REML) for the variance components. Pan and Connett (2001) [57] did not assume any parametric form for the distribution of the error term. Only a univariate random effects model was considered by Pan and Connett (2001).

There also have been Bayesian approaches to semi-parametric AFT modelling by Walker and Mallick (1999) [70] and Kottas and Gelfand (2001) [42] among others. More recently, a fully Bayesian approach to a parametric mixed effects accelerated failure time model have been suggested by Komarek, Lesaffre and Legrand (2007a) [41] and Komarek and Lesaffre (2007b) [39] using a Markov chain Monte Carlo algorithm for estimating the regression parameters with a parametric normal mixture error distribution and a multivariate normal distribution for the random effects.

Estimating a fully parametric model can be done by the usual likelihood method, i.e., by differentiating the log-likelihood. In principle, this presents no special problems in simple settings. However, some formulas get very complicated and the statistical models may not always be exponential families and therefore there is no simple way of analytical reductions [31]. As mentioned above, there have been various estimation approaches to a normal mixed effects log-linear model with correlated censored data. The most commonly applied estimation method is the EM-algorithm. Although the EM-algorithm can be computationally intensive it is a popular approach because of its conceptual simplicity and solutions can be obtained for many problems.

In this dissertation, we are interested in estimating a log-linear shared frailty AFT model when the error distribution follows a family of G^{ρ} distributions. Much of our work is motivated by the work of Vaida and Xu (2000) [69] who developed a mixed effects model approach in Cox proportional hazards model setting using a modified EM-algorithm. In the following sections, a brief review on EM-algorithm, Stochastic EM-algorithm (StEM) and Gibbs sampling is given separately.

3.1.1 Expectation-Maximization (EM)

The basic idea of EM is that one augments the observed data Y with the latent data (Z) that simplifies the calculation of the parameter estimates by performing a series of simpler maximizations or simulations. Conceptually, the EM algorithm replaces missing values by their expectations given the current parameter estimates, and then reestimates parameters using the previously estimated values of missing data, then reestimate the missing values assuming the updated parameter estimates are correct and reestimates parameters and iterates until convergence. Thus, the EM algorithm consists of two steps: Expectation (E-step) and Maximization (M-step). The E step is used to fill in the missing data. In the E-step, we take the expectation of the log of the complete data likelihood, conditioning on the observed data y and current estimates of parameters θ_n with respect to the distribution of latent data. This quantity is usually denoted as Q in the EM literature. Some authors call Q a surrogate function because Q is proportional to the log likelihood, in general [46]. The important point is that this log-augmented posterior or likelihood function should be linear in the latent data. Otherwise estimates can be severely biased when this approach is applied [65]. Let Ω = unobserved complete data, Y = observed incomplete data, Z = missing or latent data, and denote the unobserved compete data matrix as $\Omega = (Y, Z)$. Let $f(\Omega|\theta)$ be the (unobserved) complete data probability density and $p(\theta|Y, Z)$ be the augmented posterior distribution. In the E-step, we calculate the surrogate function Q:

$$\begin{aligned} Q(\theta|\theta_n) &= E[\log f(\Omega|\theta)|Y = y, \theta_n] \\ &= E_{Z|y,\theta_n}[\log f(Y, Z|\theta)|Y = y, \theta_n] \\ &= \int_Z \log[p(\theta|Y, Z)]p(Z|\theta_n, Y = y)dZ\,, \end{aligned} \tag{3.1}$$

where θ_n is the current estimated value of θ. In the M step, we maximize $Q(\theta|\theta_n)$ with respect to θ. This yields the new parameter estimate θ_{n+1}, and we repeat this two-step process until convergence.

$$|Q(\theta|\theta_{n+1}) - Q(\theta|\theta_n)| < \epsilon\,, \tag{3.2}$$

where ϵ is very small positive number. In the EM algorithm, the log-likelihood of the observed data at the parameter θ_{n+1} is always greater than or equal to the log-likelihood evaluated at the parameter θ_n. This is called the "ascent property of EM" and was proved by using measure theory and Jensen's inequality. This means that the EM algorithm increases the log-likelihood at each iteration. So, EM will converge to a parameter that maximizes the log-likelihood. However, EM may converge to a stationary point (local maxima or saddle points), rather than the global maximum. Also, EM is known for its linear convergence rate, with its rate depending on the proportion of the information about θ. This means the convergence can be slow if a large portion of the data are missing. However, EM is known for its stable and reliable convergence properties. More details on EM-algorithm can be found in Tanner and Wong (1996) and Lange (2010) [65].

3.1.2 Stochastic EM

Although the classical deterministic EM algorithm is a popular and often efficient approach to maximum likelihood estimation or for locating the posterior mode of a distribution, there are some drawbacks: (1) Its limiting (or convergence) position can depend on its starting position, (2) its rate of convergence can be slow and (3) it can provide a saddle point of the likelihood function rather than a local maximum as stated previously. In addition, the maximization step of EM is sometimes intractable. The main idea of Stochastic EM (StEM) is to impute the latent data with plausible values given the observed data and current estimates of the parameters. This is also called S-step or St-E step. Thus,

StEM incorporates a stochastic step between the E and M steps. Based on the pseudo-complete sample, compute the maximum likelihood estimates (MLE) of the parameters in the unobserved complete data log-likelihood function. The updated MLE of the parameter is stored as the new parameter value and then the process is iterated. Introducing random perturbation of the latent data in the S-step and performing subsequent maximization (the M-step) generates a Markov chain $\theta^{(m)}$ that converges to a stationary distribution under mild conditions [33]. Biscarat, Celeux and Diebolt (1992) reported that these stochastic perturbations (the random drawings) prevent the sequence $\{\theta^r\}$ from converging to the first stationary point of the log-likelihood function it encounters. At each iteration, there is a postive (a non-zero) probability of accepting an updated estimate θ^{r+1} with lower likelihood value than θ^r. Thus, Stochastic EM algorithm can avoid the saddle points or the nonsignificant local maxima of the likelihood function. In most situations, Stochastic EM yields reasonably fast convergence requiring comparatively small number of iterations. Since StEM maximizes the complete data log-likelihood of pseudocompleted samples (imputed complete data), it avoids analytic intractability that sometimes occurs in the M-step of the EM algorithm [4, 14, 33]. If one random draw is used for imputation in the St-E step it is called Stochastic EM (StEM). If more than one draw is used it is called Monte Carlo EM (MC-EM) by Wei and Tanner (1990) [71]. If the multiple draws are generated by Gibbs sampling it is called Gibbs-EM [69]. Although Gibbs-EM algorithm has a component of stochastic imputation it is different from StEM and MC-EM in that the Gibbs-EM algorithm is not necessarily maximizing the complete data log-likelihood, but rather the Q function. If it is maximizing the complete data log-likelihood it is called Data Augmentation.

3.1.3 Gibbs sampling

Gibbs sampling was originally developed by Geman and Geman (1984, 1993) as a tool for image reconstruction [22, 21]. A brief review is given in this section based on Raudenbush and Bryk (2002) [61] and Gelman et al (1995) [20]. Gibbs sampling is an approximating method for posterior distributions when they can not be evaluated analytically or have too complicated mathematical forms. By sampling from a sequence of conditional distributions, it produces draws from the approximate joint posterior at each iteration. After many iterations, the process converges stochastically, and subsequent draws may be described as representing the posterior distribution of interest.

The joint density of unknown parameters can be written as a product of conditional densities. Gibbs sampling uses this fact, capitalizing on the equivalence of different representations of the joint density. For example, when we consider a joint density with four unknowns parameters $(\mu, \beta_1, \beta_2, \alpha)$ given observed data Y and the latent data b we have

$$
\begin{aligned}
P(\mu, \beta_1, \beta_2, \alpha|Y, b) &= P_\mu(\mu|\beta_1, \beta_2, \alpha, Y, b)\, r_\mu(\beta_1, \beta_2, \alpha|Y, b) && (3.3)\\
&= P_{\beta_1}(\beta_1|\mu, \beta_2, \alpha, Y, b)\, r_{\beta_1}(\mu, \beta_2, \alpha|Y, b) && (3.4)\\
&= P_{\beta_2}(\beta_2|\mu, \beta_1, \alpha, Y, b)\, r_{\beta_2}(\mu, \beta_1, \alpha|Y, b) && (3.5)\\
&= P_\alpha(\alpha|\beta_1, \beta_2, \mu, Y, b)\, r_\alpha(\mu, \beta_1, \beta_2|Y, b), && (3.6)
\end{aligned}
$$

where $P(.)$ denotes full conditional distributions and $r(.)$ denotes densities.

We start with initial estimates $(\beta_1^{(0)}, \beta_2^{(0)}, \alpha^{(0)})$ and we sample $\mu^{(1)}$ from P_μ and then we use $(\mu^{(1)}, \beta_2^{(0)}, \alpha^{(0)})$ to sample $\beta_1^{(1)}$ from P_{β_1} and so on. This process is repeated until we reach stochastic convergence, say (m)th iteration. Usually, the joint densities $r_p(.)$ are not of importance as long as they

are free of one of the four unknown parameters ($\mu, \beta_1, \beta_2, \alpha$), for example. The empirical distributions of these samples of the unknown parameters

$$\begin{aligned}
&\mu^1 \quad ,\mu^2, \ldots, \mu^m \\
&\beta_1^1 \quad ,\beta_1^2, \ldots, \beta_1^m \\
&\beta_2^1 \quad ,\beta_2^2, \ldots, \beta_2^m \\
&\alpha^1 \quad ,\alpha^2, \ldots, \alpha^m.
\end{aligned}$$

may be regarded as an approximation to the true joint posterior distribution. Assuming (m) is large the marginal posterior for any unknown parameter may be approximated by the empirical distribution of the m samples of that unknown parameter produced by the Gibbs sampler.

3.1.3.1 ARS and ARMS Certain full conditionals can reduce analytically to well-known distributions, for which special methods for efficient random variate generation are available. However, more often, no analytical reduction is possible. For log-concave distributions efficient random variate generation can be achieved through adaptive rejection sampling (ARS) [24]. ARS works by constructing an envelope function of the log of the target density, which is then used in rejection sampling. The envelope function is a piece-wise exponential function. Whenever a point is rejected by ARS, the envelope is updated to correspond more closely to the true log density, thereby reducing the chance of rejecting subsequent points. In the original formulation of ARS, the envelope is constructed from a set of tangent lines to the log-density. The tangent line is a line that touches a curve at a point without crossing over. Formally, it is a line which which intersects a differentiable curve at a point where the slope of the curve equals the slope of the line. In a later version the envelope is constructed from chords (secants) intersecting on the log-density. Both methods assume that the log density is concave.

Gilks and Wild (1992) [24] showed that many full conditional distributions encountered in practice are log-concave. However, not all models yield log-concave full conditionals, typically in non-linear models, or with non-exponential family distributions. ARMS (Adaptive rejection metropolis sampling) deals with this situation by performing a Metropolis step on each point accepted at an ARS rejection step, suggested by Gilks, Best and Tan (1995) [23]. The Metropolis step is a Markov chain Monte Carlo (MCMC) method for obtaining a sequence of random samples from a probability distribution for which direct sampling is difficult. This sequence can be used to approximate the distribution. Details concerning the algorithms for ARS and ARMS have been published in the above mentioned references ([24, 23]).

3.2 DETAILS ON THE ESTIMATION APPROACH

A log-linear random effects AFT model is given below

$$\log T_{ij} = \beta^T Z_{ij} + b_i^T W_{ij} + \epsilon_{ij}. \tag{3.7}$$

Observed data are usually denoted as $Y = (T_{ij}, \delta_{ij}, Z_{ij}, W_{ij})$ and b_i is the vector of random effects for the ith group which follow a Gaussian distribution with mean 0 and unknown variance Σ. Then the unobserved complete data are (Y, b). β is the vector of fixed effects regression coefficients. The distribution function of the error ϵ_{ij} is denoted as $F(\epsilon; \alpha)$. Notations are summarized in the table 2.

Table 2: Notations

X_{ij}	potential event time
C_{ij}	potential censoring time
δ_{ij}	event indicator $\delta_{ij} = I(X_{ij} < C_{ij})$
T_{ij}	observed event or censoring time $T_{ij} = \min(X_{ij}, C_{ij})$
Z_{ij}	covariate vectors for the fixed effects
W_{ij}	covariate vectors for the random effects
β	a vector of fixed effects coefficients
b	a vector of random effects $(b_{0i}, b_{1i}, \ldots)$
Σ	variance-covariance matrix of b
i	cluster index $(i = 1 \ldots G)$
j	subject index $(j = 1 \ldots n_i)$

3.2.1 Complete data likelihood

In our problem, we treat the random effects b_i for ith cluster as latent or missing data. Consider the joint distribution of the random effects and the observed data y as if random effects b_i were observed. Under the following assumptions

(A.1) Censoring times are independent of survival times conditioning on the frailties b_i

(A.2) Censoring times are noninformative about random effects,

the contribution of the jth observation in the ith group to the likelihood can be written as

$$\begin{aligned} Pr[b_i, T_{ij}, \delta_{ij}] &= Pr[b_i]Pr[T_{ij}, \delta_{ij}|b_i] \\ &= Pr[b_i]Pr[T_{ij}, \delta_{ij} = 1|b_i]Pr[T_{ij}, \delta_{ij} = 0|b_i] \\ &= Pr[b_i]\Big[Pr(T_{ij} = X|\delta_{ij} = 1, b_i)Pr(\delta_{ij} = 1|b_i)\Big]\Big[Pr(T_{ij} = X|\delta_{ij} = 0, b_i)Pr(\delta_{ij} = 0|b_i)\Big] \end{aligned} \tag{3.8}$$

Then, based on the statement (3.8) unobserved complete data likelihood or full likelihood for all observations is

$$\begin{aligned} L[y, b] &= \prod_{i=1}^{G}\prod_{j=1}^{n_i} Pr[b_i]Pr[T_{ij}, \delta_{ij}|b_i] \\ &= \prod_{i=1}^{G}\left\{\prod_{j=1}^{n_i} f(t_{ij}|b_i)^{\delta_{ij}} S(t_{ij}|b_i)^{1-\delta_{ij}}\right\} Pr[b_i] \\ &= \prod_{i=1}^{G}\left\{\prod_{j=1}^{n_i} h(t_{ij}|b_i)^{\delta_{ij}} S(t_{ij}|b_i)\right\} Pr[b_i] \\ &= \prod_{i=1}^{G}\left\{\prod_{j=1}^{n_i} \left(\frac{1}{t}h_{\epsilon_{ij}}(\epsilon_{ij})\right)^{\delta_{ij}} S_{\epsilon_{ij}}(\epsilon_{ij})\right\} Pr[b_i], \end{aligned} \tag{3.9}$$

where $i = 1, 2, .., G$ clusters; $j = 1, 2, .., n_i$ subjects in cluster i. The complete data log-likelihood is

$$\begin{aligned} \log L[y, b] = l &= \sum_i \sum_j \delta_{ij} \log\left(\frac{1}{t}h_{\epsilon_{ij}}(\epsilon_{ij})\right) + \sum_i \sum_j \log S_{\epsilon_{ij}}(\epsilon_{ij}) + \sum_i \log Pr[b_i] \\ &= \left\{\sum_i \sum_j \delta_{ij}\Big\{ - \log t + \log h_{\epsilon_{ij}}(\epsilon_{ij})\Big\} + \sum_i \sum_j \log S_{\epsilon_{ij}}(\epsilon_{ij})\right\} + \sum_i \log Pr[b_i] \\ &= l_1(\beta, \alpha) + l_2(\Sigma). \end{aligned} \tag{3.10}$$

The first two terms of the above complete data log-likelihood $l_1(\beta, \alpha)$ can be rewritten by plugging in hazard $h_{\epsilon_{ij}}$ and survival $S_{\epsilon_{ij}}$ functions of error (ϵ_{ij}) random variable from (2.15) and (2.17).

$$
\begin{aligned}
l_1(\beta,\alpha) &= \sum_i \sum_j \delta_{ij} \Big\{ -\log t_{ij} + \log h_{\epsilon_{ij}}(\epsilon_{ij}) \Big\} + \sum_i \sum_j \log S_{\epsilon_{ij}}(\epsilon_{ij}) \\
&= \sum_i \sum_j \Big\{ -\delta_{ij} \log t_{ij} + \delta_{ij} \log \frac{e^{(\log t_{ij} - \beta^T Z_{ij} - b_i^T W_{ij})}}{1 + e^{\,\alpha + (\log t_{ij} - \beta^T Z_{ij} - b_i^T W_{ij})}} \Big\} \\
&- \sum_i \sum_j e^{-\alpha} \log \Big(1 + e^{\,\alpha + (\log t_{ij} - \beta^T Z_{ij} - b_i^T W_{ij})} \Big) \\
&= \sum_i \sum_j \Big\{ -\delta_{ij} \log t_{ij} + \delta_{ij} (\log t_{ij} - \beta^T Z_{ij} - b_i^T W_{ij}) - \delta_{ij} \log \Big(1 + e^{(\alpha + \log t_{ij} - \beta^T Z_{ij} - b_i^T W_{ij})} \Big) \\
&- e^{-\alpha} \log \Big(1 + e^{\,\alpha + (\log t_{ij} - \beta^T Z_{ij} - b_i^T W_{ij})} \Big) \Big\} \\
&= -\sum_i \sum_j \Big\{ \delta_{ij} (\beta^T Z_{ij} + b_i^T W_{ij}) + (\delta_{ij} + e^{-\alpha}) \log \Big(1 + e^{\,\alpha + (\log t_{ij} - \beta^T Z_{ij} - b_i^T W_{ij})} \Big) \Big\}. \qquad (3.11)
\end{aligned}
$$

And the second part of the complete data log-likelihood $l_2(\Sigma)$ can be written

$$
l_2(\Sigma) = -\frac{Gd}{2} \log(2\pi) - \frac{G}{2} \log |\Sigma| - \sum_{i=1}^{G} \frac{b_i^T \Sigma^{-1} b_i}{2}, \qquad (3.12)
$$

where we assume $b_i = (b_1, \ldots, b_d)^T$ follows multivariate normal distribution with mean zero and $d \times d$ variance-covariance matrix Σ: $f(b_i; \Sigma) = \frac{1}{(2\pi)^{d/2} |\Sigma|^{1/2}} \exp\left(-\frac{1}{2} b_i^T \Sigma^{-1} b_i\right)$ where $i = 1, 2, .., G$ clusters; $j = 1, 2, .., n_i$ subjects in cluster i; d is the number of random effects within a cluster.

3.2.2 Difficulties in the classic EM method

3.2.2.1 E-step In the E-step of a traditional EM method we calculate the expectation of the log-likelihood of the augmented data (y_i, b_i) conditional on the observed data and the current parameter

value. Let $\theta^{(k)} = (\beta^k, \alpha^k, \Sigma^k)$ be current estimates of parameter values in iterations. Then, at kth iteration for Q function in (3.1) we have

$$\begin{aligned} Q(\beta^{(k)}, \alpha^{(k)}, \Sigma^{(k)}) &= E\{l_1(\beta,\alpha)|\, Y, \theta^{(k)}\} + E\{l_2(\Sigma)|\, Y, \theta^{(k)}\} \\ &= Q_1(\beta,\alpha) + Q_2(\Sigma). \end{aligned} \tag{3.13}$$

Taking conditional expectation over complete data log likelihood for (β, α) in l_1 yields

$$\begin{aligned} Q_1(\beta,\alpha) &= -E\Big[\sum_i\sum_j \Big\{\delta_{ij}(\beta^T Z_{ij} + b_i^T W_{ij}) + (\delta_{ij} + e^{-\alpha})\log\left(1 + e^{\,\alpha + (\log t_{ij} - \beta^T Z_{ij} - b_i^T W_{ij})}\right)\Big\}|\, Y, \theta^{(k)}\Big] \\ &= -\Big\{\sum_i\sum_j \delta_{ij}(\beta^T Z_{ij}) + E[b_i^T|Y,\theta^{(k)}]W_{ij} + (\delta_{ij} + e^{-\alpha})\, E\Big[\log\left(1 + e^{\,\alpha + (\log t_{ij} - \beta^T Z_{ij} - b^T W_{ij})}\right)|\, Y, \theta^{(k)}\Big]\Big\} \\ &= -\Big\{\sum_i\sum_j \delta_{ij}(\beta^T Z_{ij}) + (\delta_{ij} + e^{-\alpha})\, E\big[g_{ij}(\beta,\alpha;b_i)|\, Y, \theta^{(k)}\big]\Big\}, \end{aligned} \tag{3.14}$$

where $g_{ij}(\beta,\alpha;b_i) = \log\left(1 + e^{\,\alpha + (\log t_{ij} - \beta^T Z_{ij} - b^T W_{ij})}\right)$. And we have Q_2

$$Q_2(\Sigma) = \sum E[\log f(b_i;\Sigma)|y,\theta_n] \propto -\frac{1}{2}\sum_{g=1}^{d}\left(G\log\sigma_g^2 + \frac{1}{\sigma_g^2}\sum_{i=1}^{G} E[b_{ig}^2|Y,\theta^{(k)}]\right). \tag{3.15}$$

This is for the multivariate normal frailty case when Σ is constrained to be a diagonal matrix. For general unconstrained Σ, a similar formula involving the expectation of cross-products $b_{ig}b_{i'g}$ can be used. Therefore, the classical E-step in EM algorithm consists of calculating the following conditional expectations $E\big[g_{ij}(\beta,\alpha;b_i)|\, Y, \theta^{(k)}\big]$ and $E[b_{ig}^2|Y,\theta^{(k)}]$. Thus, we need to calculate the conditional expectations $E[g_{ij}(\beta,\alpha;b_i)|Y,\theta^{(k)}]$ from Q_1 and $E[b_{ig}^2|Y,\theta^{(k)}]$ from Q_2, which is of the type $E[g_{ij}(\beta,\alpha;b_i)|Y,\theta^{(k)}] = \int \log\left(1 + e^{\,\alpha + (\log t_{ij} - \beta^T Z_{ij} - b^T W_{ij})}\right) p(b_{ig}|Y,\theta^{(k)})db_{ig}$ and $E[b_{ig}^2|Y,\theta^{(k)}] = \int b_{ig}^2 p(b_{ig}|Y,\theta^{(k)})db_{ig}$ for univariate random effects model. The conditional distribution of b_i given the observed data and current estimates $p(b_{ig}|Y,\theta^{(k)})$ does not have a closed form for the family of G^ρ distributions. In addition, as the dimension of random effects vector increases this integration will be multi-dimensional. For the bivariate random effects problem, Gaussian quadratures may be used [73]. For higher dimensions numerical integration becomes prohibitive [69, 73].

3.2.2.2 M-step Maximizing Q_1 in equation (3.14) is not straightforward since the parameters appear in the expectation. Our initial approach to this problem was the EM gradient algorithm by Lange (1995) [45] to linearize Q_1 in the equation (3.14) of by replacing $g_{ij}(\beta, \alpha; b_i)$ at (k+1)-th M-step with

$$G_{ij}^{(k+1)} = g_{ij}(\beta^{(k)}, \alpha^{(k)}; b_i) + (\beta - \beta^{(k)})^T \frac{\partial g_{ij}}{\partial \beta}(\beta^{(k)}, \alpha^{(k)}; b_i) + (\alpha - \alpha^{(k)})^T \frac{\partial g_{ij}}{\partial \alpha}(\beta^{(k)}, \alpha^{(k)}; b_i), \tag{3.16}$$

where the partial derivatives are

$$\begin{aligned} \frac{\partial g_{ij}}{\partial \beta}(\beta^{(k)}, \alpha^{(k)}; b_i) &= \frac{-Z_{ij}}{1 + e^{\alpha + (\log t_{ij} - \beta^T Z_{ij} - b_i^T W_{ij})}} e^{\alpha + (\log t_{ij} - \beta^T Z_{ij} - b_i^T W_{ij})} \\ &= \frac{-Z_{ij}}{e^{-\left[\alpha + (\log t_{ij} - \beta^T Z_{ij} - b_i^T W_{ij})\right]} + 1} \Big|_{\beta = \beta^{(k)}, \alpha = \alpha^{(k)}}, \end{aligned} \tag{3.17}$$

$$\begin{aligned} \frac{\partial g_{ij}}{\partial \alpha}(\beta^{(k)}, \alpha^{(k)}; b_i) &= \frac{1}{1 + e^{\alpha + (\log t_{ij} - \beta^T Z_{ij} - b_i^T W_{ij})}} e^{\alpha + (\log t_{ij} - \beta^T Z_{ij} - b_i^T W_{ij})} \\ &= \frac{1}{e^{-\left[\alpha + (\log t_{ij} - \beta^T Z_{ij} - b_i^T W_{ij})\right]} + 1} \Big|_{\beta = \beta^{(k)}, \alpha = \alpha^{(k)}}. \end{aligned} \tag{3.18}$$

and $\beta^{(k)}, \alpha^{(k)}$ are the parameter estimates from the k-th step. However, the EM algorithm failed to converge with this approximation in our problem.

3.2.3 Stochastic EM

To avoid numerical integrations in the E-step of the classical EM algorithm we propose a stochastic version of the EM algorithm (Nielsen 2000) [55] that utilizes Monte Carlo technologies such as the Gibbs sampler. In the case of no closed form for conditional distribution the Gibbs sampler may be implemented based on the adaptive rejection sampling algorithm of Gilks and Wild (1992, 1995) [24, 23]. In our problem, the M step of the EM algorithm is also not analytically tractable. Since StEM maximizes the complete data log-likelihood of pseudo completed sample it does not involve such difficulties. The basic idea underlying Stochastic EM (StEM) is to replace the computation and maximization of the Q function by the simpler computation and simulation of an unobserved random

effects b_i, and then to update $\theta^{(k)} = (\beta^{(k)}, \alpha^{(k)})$ on the basis of the pseudocompleted sample $x^c = (y, b_i^c)$. More specifically, we iterate the following two procedures with an initial guess of $\theta^{(0)}$.

(1) Stochastic E-step (S-step): draw Monte Carlo samples from the conditional distribution of the latent data given the observed data and $\theta^{(0)}$ to form a pseudo-complete sample.

(2) M-step: find the maximum likelihood estimator (MLE) $\theta^{(k)}$ based on the pseudo-complete data, and update $\theta^{(k)}$ to $\theta^{(k+1)}$.

The full conditional distribution of the latent data in S-step for our problem is given by

$$p(b_{ig}|\mathbf{b}_{i(-g)}, \mathbf{y}_i, \theta^{(k)}),$$

where $\mathbf{b}_{i(-g)} = (b_{i1}, \ldots, b_{i,k-1}, b_{i,k+1}, \ldots, b_{id})$. And the Gibbs sampler proceeds by successively sampling from $p(b_{ig}|\mathbf{b}_{i(-g)}, \mathbf{y}_i, \theta^{(k)})$ for $g = 1 \ldots d$, using the adaptive rejection sampling algorithm [24, 23].

Imputing pseudo-complete data (the S-step) and subsequent maximization (the M-step) alternately generates a Markov chain $\{\theta^{(m)}\}$ that converges to a stationary distribution Ψ under mild conditions [33]. Thus, the Stochastic EM point estimate for θ is

$$Mean(\Psi) = \frac{1}{M - m_0} \sum_{m=m_0+1}^{M} \theta^{(m)}, \tag{3.19}$$

where M is a total number of draws or EM iterations and m_0 is a burn-in period to approximately reach the stationarity and is discarded from the calculation. It has been shown that its relationship to MLE $(\hat{\theta})$ is $Mean(\Psi) = \hat{\theta} + O(1/n)$ for the exponential family case.

The maximum likelihood estimator (MLE) of Q_1 in the M-step can be found by maximizing the complete data log-likelihood via a non-linear optimization method.

$$\theta_{n+1} \cong \arg\max_{\theta} l_1(Y, b_n|\theta_n).$$

This is very straightforward step now due to the imputation. To maximize Q_2, Vaida and Xu (2000) noted that this is the log-likelihood corresponding to G independent observations from the 'prior' random effects distribution $p(\mathbf{b}_i)$ in which the standard sufficient statistics are replaced with their conditional expectations and, in general, the solution is readily available. That is, in the case of diagonal G the estimates are

$$\widehat{\sigma}_g^2 = \frac{1}{G}\sum_{i=1}^{G} E[b_{ig}^2|Y, \theta^{(k)}], \tag{3.20}$$

for $g = 1, \ldots, d$. In the Stochastic EM, the expectations $E[b_{ig}^2|Y, \theta^{(k)}]$ are replaced by a single Monte Carlo draw for each b_{ig}. If the variance matrix Σ is unconstrained, then it is maximized by

$$\widehat{\Sigma} = \frac{1}{G}\sum_{i=1}^{G} E[b_i b_i^T|Y, \theta^{(k)}], \tag{3.21}$$

where $b_i = [b_{i1}, \ldots, b_{ig}]^T$ and g is the dimension of random effects.

3.2.4 Variance estimation

The observed information matrix $I(\hat{\theta})$ can be computed from the complete data log-likelihood function denoted by ℓ^c using the Louis' method [50].

$$I(\hat{\theta}) = -E[\nabla^2 \ell^c(\theta; y, b)|y, \theta] - E[\nabla \ell^c(\theta; y, b)^{\otimes 2}|y, \theta], \tag{3.22}$$

where $u^{\otimes 2} = uu^T$, ∇ and ∇^2 denote the first and the second derivatives with respect to parameters. The expectations in the above Louis formula is computed by an empirical version in Stochastic EM:

$$\frac{1}{M - m_0}\sum_{i=m_0+1}^{M} -\nabla^2 \ell^c(\theta; y, b^{(i)}) - \frac{1}{M - m_0}\sum_{i=m_0+1}^{M} \nabla \ell^c(\theta; y, b^{(i)})^{\otimes 2}, \tag{3.23}$$

where the parameter θ is fixed at $Mean(\Psi_n)$. M and m_0 are defined earlier. The components of $\nabla \ell^c(b)$, the first derivatives of complete data log-likelihood, are

$$\frac{\partial \ell}{\partial \beta} = (-1) \sum_i^G \sum_j^{n_i} \frac{Z_{ij}(\delta_{ij} - e^{-\alpha} e^A)}{1 + e^A}, \tag{3.24}$$

$$\frac{\partial \ell}{\partial \alpha} = (-1) \sum_i^G \sum_j^{n_i} \left\{ - e^{-\alpha} \log(1 + e^A) + \frac{e^A(\delta_{ij} + e^{-\alpha})}{1 + e^A} \right\}, \tag{3.25}$$

$$\frac{\partial \ell}{\partial \sigma_g^2} = -\frac{1}{2} \left\{ \frac{G}{\sigma_g^2} - \frac{\sum_i^G b_{ig}^2}{(\sigma_g^2)^2} \right\}, \tag{3.26}$$

for $g = 1, \ldots, d$ diagonal case and ℓ is given in equation (3.10). For the second derivatives

$$\frac{\partial^2 \ell}{\partial \beta^2} = (-1) \sum_i^G \sum_j^{n_i} \frac{Z_{ij}^{\otimes 2} e^A (e^{-\alpha} + \delta_{ij})}{(1 + e^A)^2}, \tag{3.27}$$

$$\frac{\partial^2 \ell}{\partial \alpha^2} = (-1) \sum_i^G \sum_j^{n_i} \left\{ e^{-\alpha} \log(1 + e^A) - \frac{e^A e^{-\alpha}}{1 + e^A} + \frac{e^A(\delta_{ij} - e^{-\alpha} e^A)}{(1 + e^A)^2} \right\}, \tag{3.28}$$

$$\frac{\partial^2 \ell}{\partial (\sigma_g^2)^2} = -\frac{1}{2} \left\{ \frac{G}{(\sigma_g^2)^2} - \frac{\sum_i^G b_{ig}^2}{(\sigma_g^2)^3} \right\}, \tag{3.29}$$

$$\frac{\partial^2 \ell}{\partial \beta \partial \alpha} = (-1) \sum_i^G \sum_j^{n_i} \frac{-e^A Z_{ij}(\delta_{ij} - e^{-\alpha} e^A)}{(1 + e^A)^2}, \tag{3.30}$$

where $e^A = e^{\alpha + (\log t_{ij} - \beta^T Z_{ij} - b_i^T W_{ij})}$ and $z^{\otimes 2} = z z^T$ for vector z, and all other off-diagonal elements of $\nabla^2 \ell_{ij}$ are zero.

3.2.5 Inference

A Wald-type statistic can be used to test whether each of the fixed effects parameter estimates differs from a specified value. Inference regarding the inclusion or exclusion of random effects in linear mixed models is challenging because the variance components are located on the boundary of their parameter space under the usual null hypothesis of the variance components being zero. As a result, the asymptotic null distribution of the Wald, score, and likelihood ratio tests will not have the χ^2 distribution. Especially, the null distribution of the likelihood ratio test (LRT) is shown to follow a 50:50 mixture χ^2 distribution (Self and Liang 1987 [62], Stram and Lee 1994 [64], Zhang and Lin 2008 [74]). The p-value of the LRT test for given observed LRT statistic T_{obs} is given as

$$0.5[\chi^2_s \geq T_{obs}] + 0.5P[\chi^2_{s+1} \geq T_{obs}]\,, \tag{3.31}$$

where $T_{obs} = -2\log L(\hat{\beta}, \hat{\alpha}) + 2\log L(\hat{\beta}, \hat{\alpha}, \hat{\theta})$ and s is the number of variance components in Σ. For example, $s = 1$ for univariate shared AFT frailty model. $s = 2$ for bivariate shared AFT frailty model in the absence of correlation.

As for model selection criteria, the classical Akaike Information Criterion (AIC) (Akaike 1973) [2] for parametric models is formulated by rewarding goodness of fit via $-2\log L$ but penalizing the number of estimated parameters, p, in the model by adding $2p$.

$$AIC = -2\log L + 2p. \tag{3.32}$$

In the parametric AFT model setting, p includes the number of parameters associated with the error distribution such as shape and scale parameters. In the presence of random effects, Vaida and Blanchard (2003) [68] distinguished between the marginal and the conditional focus of comparisons in the linear random effects model context. When we are interested in the inference on population parameters but not on the random effects and these random effects are only a convenient way of modeling

the correlation within a cluster and the real interests lie on the fixed effects parameters, the marginal Akaike Information Criterion (mAIC) (Burnham and Anderson 2002) [6] is used.

$$mAIC = -2\log L + 2(p+s), \tag{3.33}$$

where s is the number of variance components and p is as defined earlier. Whereas the random effects themselves are quantities of interest Vaida and Blanchard (2003) [68] proposed a conditional Akaike Information Criterion (cAIC).

$$cAIC = -2\log L + 2(\rho^* + 1), \tag{3.34}$$

where ρ^* is the effective degrees of freedom and is estimated by $\rho^* = trace(H)$ where H is the pseudo-projection matrix [68]. There is currently no literature of application of cAIC in the AFT frailty model setting. For all AIC calculations across different models it is important to keep the number of observations equal since each observation contributes to the log-likelihood.

4.0 SIMULATION

Simulation studies were conducted to evaluate the performance of finite sample properties of the proposed estimation procedure for the log-linear AFT model with a family of G^{ρ} distribution in both the univariate and bivariate shared frailty model setting. We also examine the robustness of the estimated fixed effects and the estimated variance components when the error distribution or distribution of the random effects is misspecified. Performance of the estimators are evaluated by the following measures. The percentage bias is computed as proportion of the difference from the true value:

$$\left(\frac{\hat{\beta_N} - \beta}{\beta}\right) * 100, \tag{4.1}$$

where β is a true value and the mean parameter estimate is

$$\hat{\beta_N} = \frac{1}{N}\sum_{1}^{N}\hat{\beta}_i, \tag{4.2}$$

where N is a total number of simulation datasets generated and $\hat{\beta}_i = \frac{1}{M-m_0}\sum_{m=m_0+1}^{M}\theta_i^{(m)}$ from equation (3.19) in section 3.2.3. By the definition in equation (4.1) positive percent bias indicates over-estimation and negative percent bias indicates under-estimation of the true parameter. The sample standard deviation of the parameter estimators (SD) is the empirical standard error computed as

$$SD(\hat{\beta}) = \sqrt{\frac{1}{N-1}\sum_{1}^{N}(\hat{\beta}_i - \hat{\beta_N})^2}. \tag{4.3}$$

The standard error (SE) of the parameter estimate is the average of the model-based estimates of the large sample standard errors for the parameter estimate by the Louis method [50] as indicated in

equation (3.23) in section 3.2.4. The mean squared error (MSE) of the estimated regression coefficients is defined as

$$\widehat{MSE} = \frac{1}{N}\sum_{i=1}^{N}(\hat{\beta}_i - \beta)^2 \approx (\hat{\beta_N} - \beta)^2 + (SE(\hat{\beta}))^2, \tag{4.4}$$

where $SE(\hat{\beta})$ is the average large sample standard error of the estimate over all simulations. The 95% coverage probability (CP) is the observed proportion of simulations for which the 95% confidence interval includes the true value. For all data simulation and parameter estimation, R software was used [60].

4.1 SIMULATION MODEL I

4.1.1 Shared frailty model in univariate setting

Simulation datasets were generated from the following log-linear G^ρ family AFT model with random effects.

$$\log T_{ij} = \beta_0 + \beta_1 x_{1ij} + \beta_2 x_{2ij} + b_i + \tau * \epsilon(\alpha)_{ij}\,, \tag{4.5}$$

where $i = 1...G$ clusters, $j = 1...n_i$ subjects within a cluster. The covariates x_{1ij} and x_{2ij} follow normal(0,1) and I(binomial(0.5) $>$ 0.5) distributions, respectively. The random effects $\mathbf{b} = (b_1, \ldots, b_i, \ldots, b_G)$ are assumed to follow a normal distribution, N(0, θ). τ is the scale parameter. The error term ϵ follows the G^ρ family distribution with an additional parameter α which reflects the degree of the attenuation of the hazard functions.

The random cluster effect, b_i, is shared by members within each cluster but varies independently across clusters. An example of this setting is a multi-center clinical trial with centers as clusters. Center effects are regarded as random and one failure time is recorded for each patient within each

center. For simulations, the **b** were independently drawn from a normal distribution with mean 0 and variance θ. This θ represents the variance component of univariate random effects model. The fixed effect covariates in the simulation are x_{1ij} and x_{2ij}. The continuous covariate x_{1ij} can be considered to be some baseline measurement for which adjustment is needed for. The binary covariate x_{2ij} can be considered as a treatment assignment, e.g., a clinical trial in which subjects were randomly assigned to one of two treatments within each cluster. β_0 represents the intercept term which is the log baseline survival time. The scale parameter τ is fixed at 1 and not estimated in simulations.

The true values of the coefficients corresponding to the covariate vector $x^T = (1, x_{1ij}, x_{2ij})$ were $\beta^T = (\beta_0, \beta_1, \beta_2) = (1, 1, 1)$. The α was arbitrarily set at 1. The variance component θ of the random effects was set at 1. The following number of clusters were considered in the univariate shared frailty model setting: $G = (10, 20, 50, 100)$ with the number of observations per cluster (i.e. cluster size) varying from 8 to 20, $n = (8, 12, 20)$. The error (ϵ_{ij}) followed the G^ρ family distribution where the parameter ρ is re-parametrized as $\alpha = \log \rho$. The four different values of $\alpha = \log \rho = (-0.69, 0.60, 0.00, 1.00)$ were evaluated in the simulated models as well. Of note, $\alpha = 0$ indicates a proportional odds model. Censoring times were generated independently from the exponential distribution with the parameter that resultsed in the censoring proportion of approximately 20%.

4.1.2 Simulation Results

Figure (4.1) shows 2000 StEM sequences for the model parameters after fitting an AFT shared frailty model $Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + b_i + \epsilon$ with G=50, n=20, $(\beta_0, \beta_1, \beta_2, \alpha) = (1.0, 1.0, 1.0, 1.0)$ and $\theta = 1.0$ for the case of 20% censoring. Convergence plots in panel (a) show the magnification of the first 100 StEM sequences. Panel (b) shows the convergence plots with the full 2000 StEM sequences. All parameter estimates converged quickly except for the intercept term μ, but the difference in the

number of iterations to reach the stationary was less than 20. Whether we used 2000 or 250 Stochastic EM iterations the results did not change much, therefore all simulations from the univariate random effects model were performed with m =250 Stochastic EM runs. In addition, the burn-in period was set at $m_0 = 50$ to reach the stationary sequence θ^m.

The mean parameter estimates (Mean), empirical standard errors (SD), asymptotic standard errors (SE), percentage bias, mean squared error (MSE) and 95% coverage rate are reported in Tables 3 through 6 for simulation datasets with different numbers of clusters $G = (10, 20, 50, 100)$ and cluster sizes $n = (8, 12, 20)$ using the univariate shared AFT frailty model specified earlier in the simulation model equation (5.1).

We examined the effects of number of clusters (Table 3), cluster size (Table 4 and 5), and the effect of changes in the parameter α (Table 6). Both fixed effects and random effect parameters are estimated with as few as $G = 20$ clusters and $n = 20$ observations with percentage bias less than or about 3%. An increase in the number of clusters reduces standard errors and increases precision of the estimates. The percentage bias of the random effect estimate is noticeably reduced with more clusters when the number of observations is fixed (Figure 4.3).

The asymptotic standard errors (SE) are not perfectly aligned with empirical standard errors (SD). Nonetheless, as the number of cluster increases the asymptotic standard errors appear to be closer to the empirical standard errors. But there is noticeable underestimation of the empirical standard errors of the intercept parameter β_0 (Table 3). When the number of clusters is fixed the effects of cluster size has the largest impact on the random effect parameter estimation (Figure 4.4).

The larger the cluster size the more precise the mean estimate is (Table 4). In addition, an increase in the number of clusters when the cluster size is small does not improve the estimates of random effects. For example, when the number of clusters is increased from $G = 20$ to $G = 50$ but the cluster

size is only $n = 8$ the percentage bias for the random effect estimate is around 50% (Table 4 and Table 5). This indicates that enough observations are needed in each cluster to reduce the bias of estimation of random effects. When the parameter α is varied from $\alpha = (-0.69, 0.60, 0.00, 1.00)$ all parameter estimation is reasonably good with percent bias less than 3.5% for all parameters and less than 2.1% for the parameter α (Table 6).

As a result of the last Stochastic E-step we also obtain the predicted random effects for each cluster, that is $\hat{b_i} = E[b_i|y, \hat{\theta}]$ for $i = 1, \ldots, G$, with variance $\hat{v_i} = var(b_i|y_i, \hat{\theta})$ and the corresponding 95 per cent credibility intervals (Bayesian confidence intervals) using the normal approximation i.e., $(E[b_i|y, \hat{\theta}] \pm 2\hat{v_i^{1/2}})$. These results are shown in Figure (4.5). Observed and predicted random effects are plotted based on the results after fitting an AFT shared frailty model $Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + b_i + \epsilon$ with G=50, n=20, $(\beta_0, \beta_1, \beta_2, \alpha, \theta) = (1.0, 1.0, 1.0, 1.0, 1.0)$ for the case with 20% censoring. In general, the predicted random effects are well within the credibility intervals.

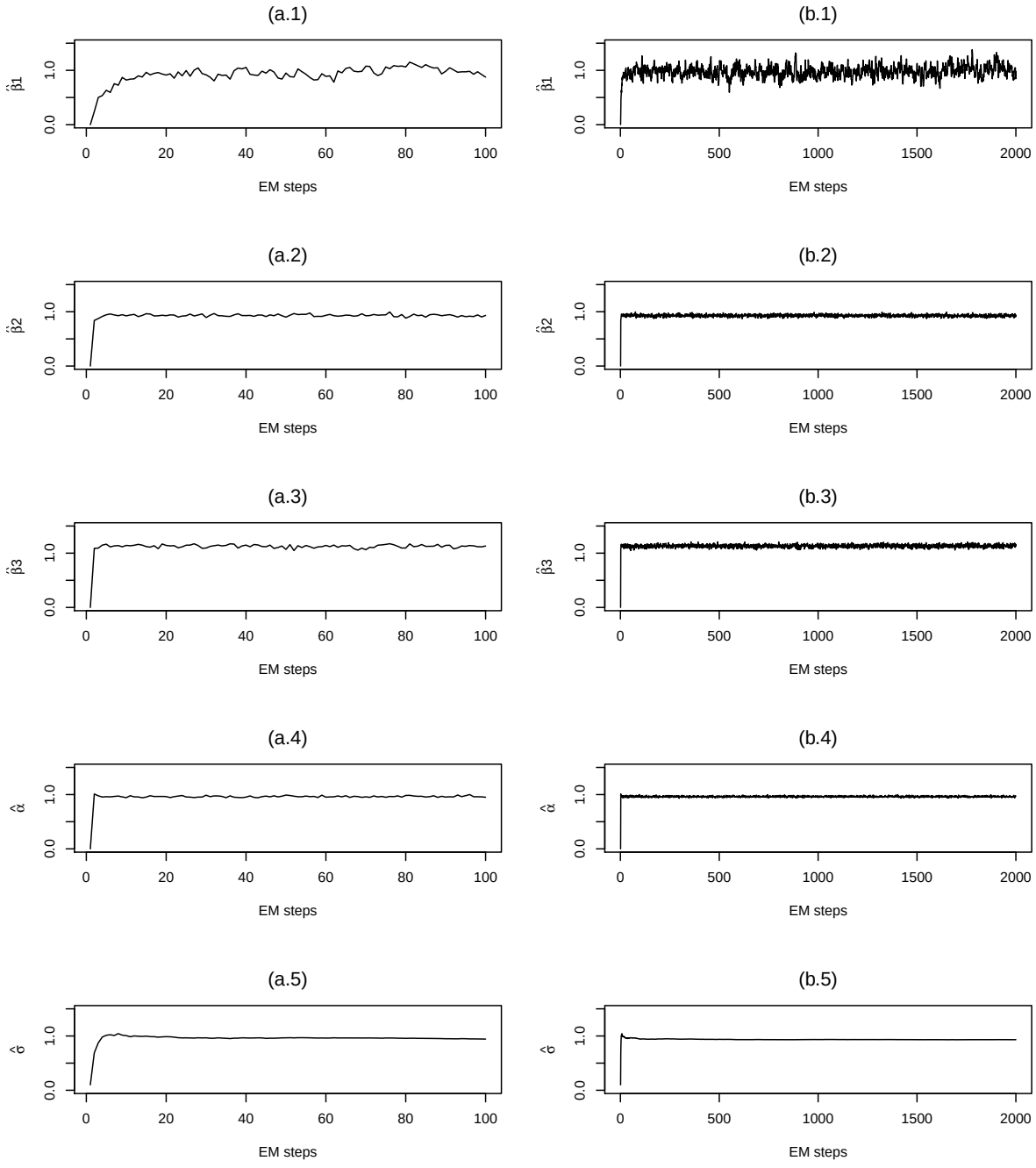

Figure 4.1: StEM sequences for the model parameters

Figure (4.2) shows the histograms with smoothed density curves (dotted line) and normal density overlay (solid line) of the parameter estimates from 200 simulated datasets based on an AFT shared frailty model $Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + b_i + \epsilon$ with G=100, n=20, $(\beta_0, \beta_1, \beta_2, \alpha, \theta) = (1, 1, 1, 1, 1)$ for the case of 20% censoring. These histograms show that the estimated regression parameters are approximately symmetrically distributed.

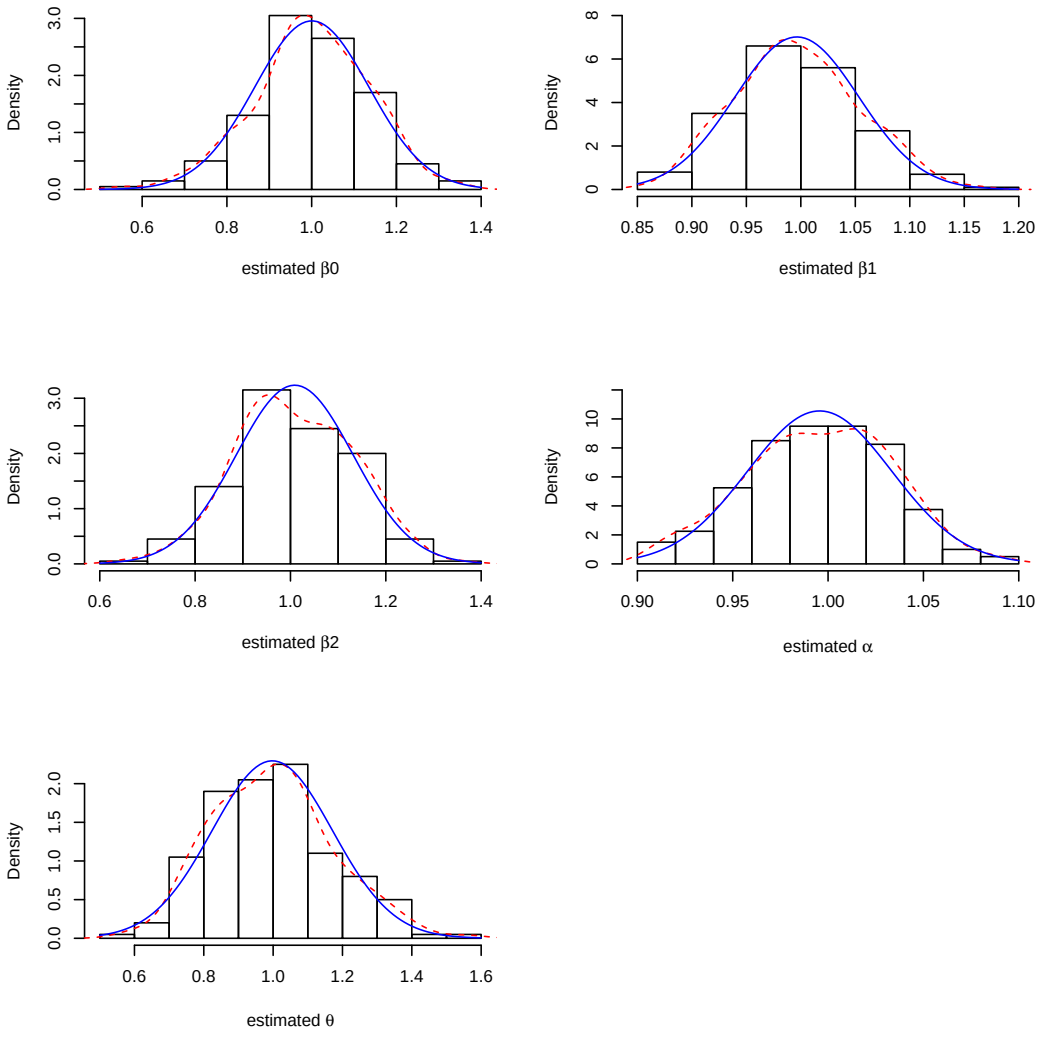

Figure 4.2: Distribution of estimated model parameters

Table 3: Effects of number of clusters

	Parameter	True Value	Mean	SD	SE	Bias (%)	MSE	coverage (%)
G=10								
	β_0	1	0.940	0.435	0.188	-6.0	0.192	58.0
	β_1	1	0.997	0.188	0.190	-0.3	0.035	94.0
	β_2	1	0.975	0.365	0.272	-2.5	0.133	84.0
	α	1	0.999	0.138	0.133	-0.1	0.019	94.5
	θ	1	0.907	0.511	0.457	-9.3	0.268	75.0
G=20								
	β_0	1	1.020	0.305	0.128	2.0	0.093	52.5
	β_1	1	1.001	0.136	0.133	0.1	0.018	94.0
	β_2	1	0.997	0.284	0.188	-0.3	0.080	80.0
	α	1	0.987	0.086	0.094	-1.3	0.007	97.0
	θ	1	0.985	0.383	0.346	-1.5	0.146	85.0
G=50								
	β_0	1	1.031	0.174	0.080	3.1	0.031	66.0
	β_1	1	1.003	0.082	0.084	0.3	0.007	94.5
	β_2	1	0.981	0.171	0.118	-1.9	0.029	82.0
	α	1	0.992	0.052	0.059	-0.8	0.003	95.0
	θ	1	1.017	0.238	0.225	1.7	0.057	93.0
G=100								
	β_0	1	1.000	0.135	0.057	0.0	0.018	60.0
	β_1	1	0.996	0.057	0.060	-0.4	0.003	97.0
	β_2	1	1.009	0.123	0.084	0.9	0.015	83.0
	α	1	0.996	0.038	0.042	-0.4	0.001	97.0
	θ	1	0.998	0.174	0.160	-0.2	0.030	90.5

Table 4: Effects of cluster size: G=20

	Parameter	True Value	Mean	SD	SE	Bias (%)	MSE	coverage (%)
n=8								
	β_0	1	0.972	0.384	0.257	-2.8	0.147	78.5
	β_1	1	0.987	0.215	0.217	-1.3	0.046	95.5
	β_2	1	1.043	0.390	0.335	4.3	0.153	90.0
	α	1	0.982	0.147	0.159	-1.8	0.022	97.0
	θ	1	0.486	0.354	0.172	-51.4	0.389	26.5
n=12								
	β_0	1	1.003	0.366	0.188	0.3	0.133	62.0
	β_1	1	1.014	0.161	0.176	1.4	0.026	97.5
	β_2	1	0.979	0.317	0.261	-2.1	0.100	88.5
	α	1	0.986	0.128	0.126	-1.4	0.017	95.0
	θ	1	0.785	0.385	0.275	-21.5	0.194	63.0
n=20								
	β_0	1	1.020	0.305	0.128	2.0	0.093	52.5
	β_1	1	1.001	0.136	0.133	0.1	0.018	94.0
	β_2	1	0.997	0.284	0.188	-0.3	0.080	80.0
	α	1	0.987	0.086	0.094	-1.3	0.007	97.0
	θ	1	0.985	0.383	0.346	-1.5	0.146	85.0

Table 5: Effects of cluster size: G=50

	Parameter	True Value	Mean	SD	SE	Bias (%)	MSE	coverage (%)
n=8								
	β_0	1	0.933	0.247	0.158	-6.7	0.065	79.5
	β_1	1	0.998	0.138	0.141	-0.2	0.019	94.0
	β_2	1	0.995	0.229	0.211	-0.5	0.052	91.5
	α	1	1.022	0.089	0.097	2.2	0.008	96.5
	θ	1	0.500	0.240	0.110	-50.0	0.308	16.0
n=12								
	β_0	1	0.969	0.199	0.116	-3.1	0.041	70.0
	β_1	1	1.017	0.100	0.112	1.7	0.010	96.5
	β_2	1	1.012	0.206	0.161	1.2	0.043	87.0
	α	1	0.997	0.080	0.079	-0.3	0.006	95.0
	θ	1	0.806	0.237	0.176	-19.4	0.093	64.5
n=20								
	β_0	1	1.031	0.174	0.080	3.1	0.031	66.0
	β_1	1	1.003	0.082	0.084	0.3	0.007	94.5
	β_2	1	0.981	0.171	0.118	-1.9	0.029	82.0
	α	1	0.992	0.052	0.059	-0.8	0.003	95.0
	θ	1	1.017	0.238	0.225	1.7	0.057	93.0

Table 6: Effects of α

	Parameter	True Value	Mean	SD	SE	Bias (%)	MSE	coverage (%)
$\alpha = -0.69$								
	β_0	1.00	1.014	0.158	0.051	1.4	0.025	50.0
	β_1	1.00	0.999	0.052	0.052	-0.1	0.003	96.5
	β_2	1.00	0.998	0.096	0.073	-0.2	0.009	85.0
	α	-0.69	-0.704	0.142	0.160	2.1	2.924	98.0
	θ	1.00	0.966	0.226	0.203	-3.4	0.052	87.5
$\alpha = 0.60$								
	β_0	1.00	1.021	0.173	0.068	2.1	0.030	54.0
	β_1	1.00	1.007	0.072	0.072	0.7	0.005	94.5
	β_2	1.00	1.008	0.133	0.101	0.8	0.018	84.5
	α	0.60	0.595	0.065	0.069	-0.8	0.168	95.0
	θ	1.00	0.983	0.243	0.215	-1.7	0.059	89.0
$\alpha = 0.00$								
	β_0	1.00	1.001	0.178	0.057	0.1	0.031	43.0
	β_1	1.00	0.999	0.058	0.059	-0.1	0.003	94.5
	β_2	1.00	1.003	0.111	0.084	0.3	0.012	84.5
	α	0.00	-0.016	0.087	0.097	-1.6	1.041	98.5
	θ	1.00	0.979	0.243	0.209	-2.1	0.059	84.5
$\alpha = 1.00$								
	β_0	1.00	1.031	0.174	0.080	3.1	0.031	66.0
	β_1	1.00	1.003	0.082	0.084	0.3	0.007	94.5
	β_2	1.00	0.981	0.171	0.118	-1.9	0.029	82.0
	α	1.00	0.992	0.052	0.059	-0.8	0.003	95.0
	θ	1.00	1.017	0.238	0.225	1.7	0.057	93.0

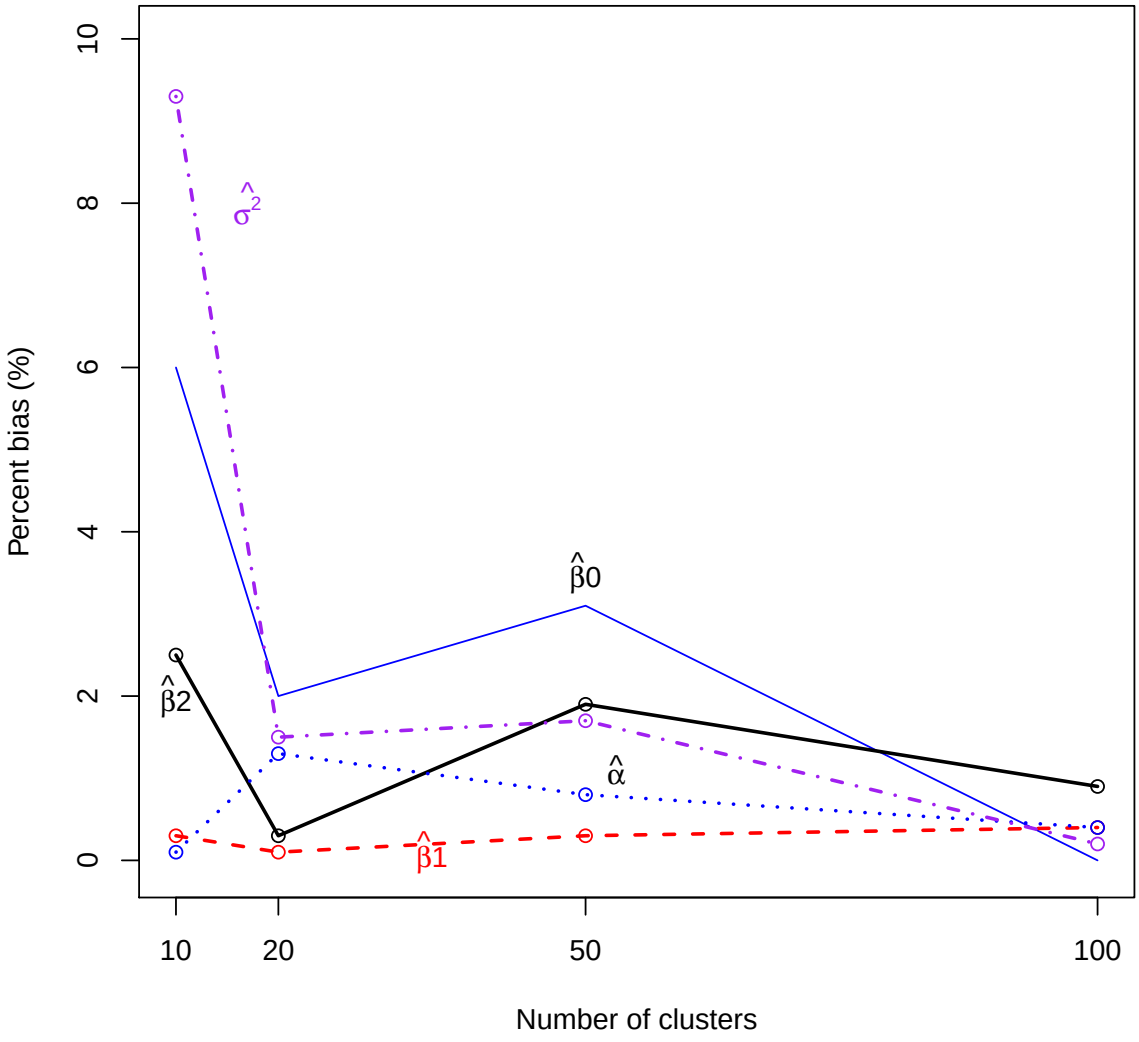

Figure 4.3: Number of clusters and percent bias
with n=20, $(\beta_0, \beta_1, \beta_2, \alpha, \theta) = (1, 1, 1, 1, 1)$

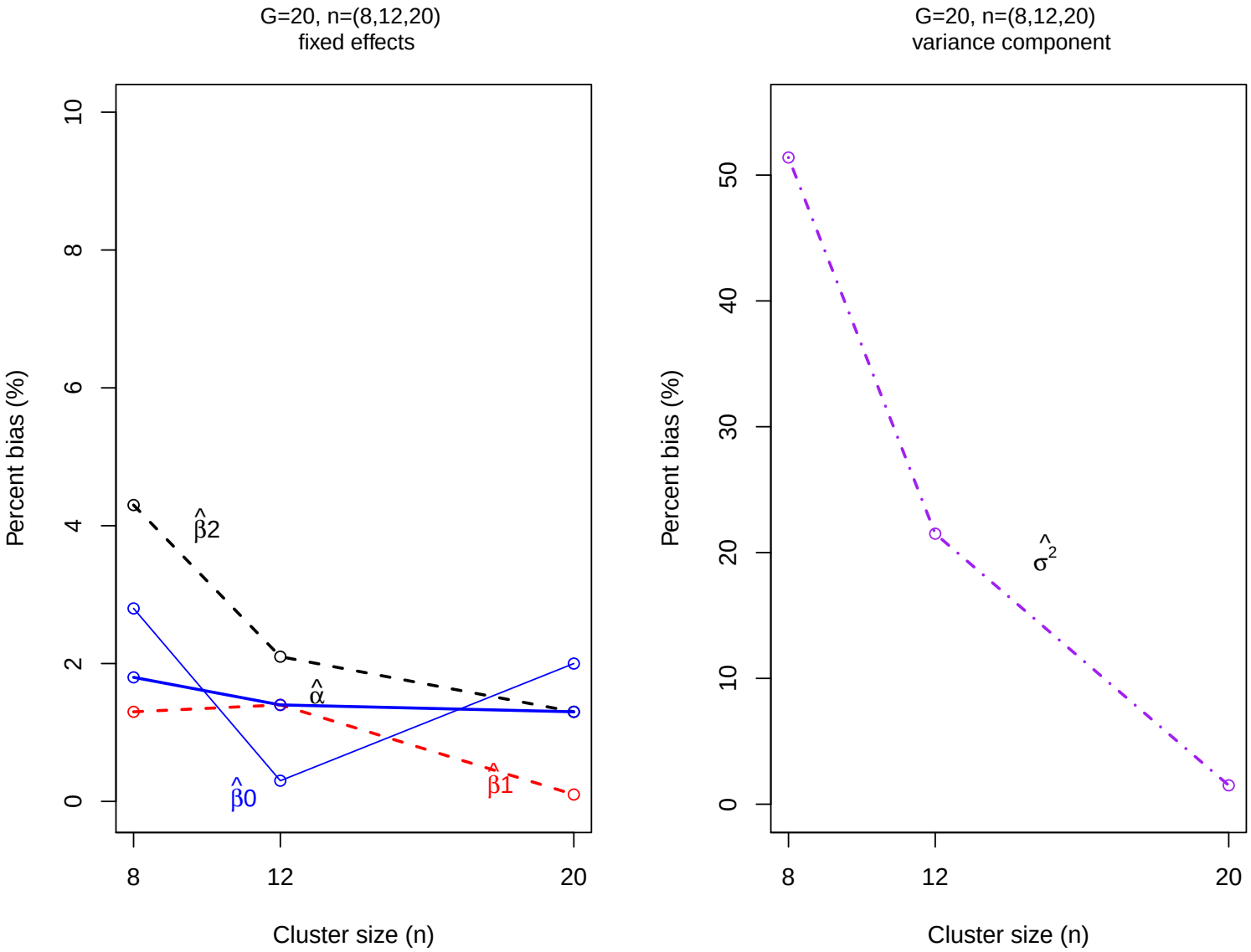

Figure 4.4: Cluster sizes and percent bias

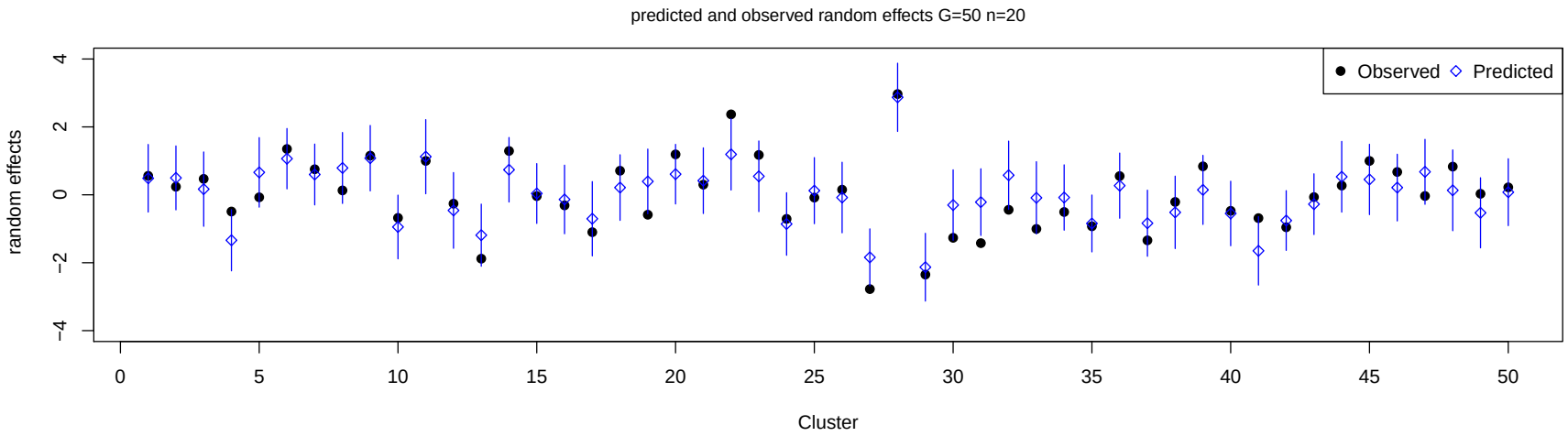

Figure 4.5: Observed and predicted random effects

4.1.3 Misspecified models

In this section, we present the impact of misspecifying the error distribution and the frailty distribution on mean parameter estimates (Mean), empirical standard errors (SD), asymptotic standard errors (SE), percentage bias, mean squared error (MSE) and coverage rate in the univariate AFT shared frailty model. Simulation datasets were generated from the following log-linear G^{ρ} family AFT model with random effects.

$$\log T_{ij} = \beta_0 + \beta_1 x_{1ij} + \beta_2 x_{2ij} + b_i + \epsilon(\alpha)_{ij}\,, \tag{4.6}$$

where $i = 1...G$ clusters, $j = 1...n_i$ subjects within a cluster. The covariates x_{1ij} and x_{2ij} follow normal(0,1) and I(binomial(0.5) > 0.5) distributions, respectively. True values of the parameters are set at $(\beta_0, \beta_1, \beta_2) = (1, 1, 1)$ but the true value of the parameter α can not be set at a predetermined value for the misspecified error model. This is because the parameter α is specific to the G^{ρ} distribution. The following distributions are considered for e_{ij} and b_i.

Misspecified error model (e_{ij}): We considered a standard log normal distribution and standard logistic distribution as the distribution functions of the error $F(\epsilon; \alpha)$ misspecification. For both log normal and logistic distribution the percentage bias of covariate effects for (β_1, β_2) are all less than 1%. The variance components (θ) of the random effects (b_i) are overestimated in general with log normal error distribution. In addition, the bias of the intercept term β_0 is fairly large (Table 7). The model performs well with the logistic error distribution (Table 8 and Table 9).

Misspecified frailty model (b_i): Gamma distribution, Inverse Gaussian distribution and log normal distribution are considered for the distribution of misspecified random effects b_i. For all mis-

specified frailty distributions the percentage bias of covariate effects for (β_1, β_2) is less than 1%. The variance components θ of random effects b_i are also well estimated with less than 5% percentage bias (Table 10). Overall, the results indicate that misspecified AFT models give asymptotically unbiased estimates of the covariate effect, but the estimate of the intercept parameter β_0 is biased.

Table 7: Effects of standard log normal misspecified error distribution

	Parameter	True Value	Mean	SD	SE	Bias (%)	MSE	coverage (%)
$n = 30$								
	β_0	1.00	1.386	0.175	0.034	38.6	0.180	3.5
	β_1	1.00	1.005	0.036	0.038	0.5	0.001	96.0
	β_2	1.00	1.009	0.075	0.052	0.9	0.006	83.5
	α	NA	-4.612	4.209	31.810	-.-	-.-	-.-
	θ	1.00	1.064	0.265	0.262	6.4	0.074	97.0
$n = 40$								
	β_0	1.00	1.333	0.165	0.030	33.3	0.138	3.5
	β_1	1.00	1.010	0.029	0.033	1.0	0.001	95.0
	β_2	1.00	1.006	0.059	0.045	0.6	0.004	87.0
	α	NA	-2.780	1.782	3.338	-.-	-.-	-.-
	θ	1.00	1.135	0.275	0.293	13.5	0.093	96.5
$n = 50$								
	β_0	1.00	1.279	0.184	0.027	27.9	0.112	5.5
	β_1	1.00	1.008	0.029	0.030	0.8	0.001	94.0
	β_2	1.00	1.008	0.052	0.040	0.8	0.003	87.0
	α	NA	-2.581	1.242	1.895	-.-	-.-	-.-
	θ	1.00	1.216	0.272	0.323	21.6	0.120	99.0

Table 8: Effects of standard logistic misspecified error distribution

	Parameter	True Value	Mean	SD	SE	Bias (%)	MSE	coverage (%)
$n = 20$								
	β_0	1.00	1.002	0.260	0.092	0.2	0.067	49.0
	β_1	1.00	0.995	0.102	0.094	-0.5	0.010	94.5
	β_2	1.00	1.006	0.178	0.134	0.6	0.032	86.5
	α	0.00	-0.013	0.130	0.155	-1.3	1.043	98.0
	θ	1.00	0.963	0.329	0.330	-3.7	0.109	87.0
$n = 30$								
	β_0	1.00	0.997	0.265	0.070	-0.3	0.070	44.0
	β_1	1.00	1.010	0.078	0.076	1.0	0.006	93.5
	β_2	1.00	0.977	0.160	0.106	-2.3	0.026	78.5
	α	0.00	-0.006	0.107	0.121	-0.6	1.024	98.0
	θ	1.00	1.017	0.366	0.352	1.7	0.134	91.0
$n = 40$								
	β_0	1.00	0.961	0.226	0.059	-3.9	0.053	35.0
	β_1	1.00	1.002	0.065	0.066	0.2	0.004	96.0
	β_2	1.00	1.012	0.116	0.090	1.2	0.014	87.0
	α	0.00	-0.001	0.096	0.104	-0.1	1.011	97.0
	θ	1.00	1.061	0.350	0.372	6.1	0.126	94.0

Table 9: Effects of standard logistic misspecified error distribution

	Parameter	True Value	Mean	SD	SE	Bias (%)	MSE	coverage (%)
$n = 20$								
	β_0	1.00	0.985	0.196	0.063	-1.5	0.038	48.0
	β_1	1.00	1.003	0.064	0.066	0.3	0.004	94.0
	β_2	1.00	1.002	0.137	0.093	0.2	0.019	82.0
	α	0.00	-0.009	0.100	0.107	-0.9	1.029	95.5
	θ	1.00	0.981	0.248	0.234	-1.9	0.061	88.0
$n = 30$								
	β_0	1.00	0.984	0.162	0.049	-1.6	0.026	43.0
	β_1	1.00	1.004	0.048	0.054	0.4	0.002	95.5
	β_2	1.00	1.001	0.105	0.075	0.1	0.011	81.5
	α	0.00	-0.007	0.073	0.086	-0.7	1.019	97.0
	θ	1.00	1.023	0.257	0.247	2.3	0.066	94.0
$n = 40$								
	β_0	1.00	1.007	0.172	0.042	0.7	0.030	23.0
	β_1	1.00	0.987	0.037	0.046	-1.3	0.002	97.5
	β_2	1.00	1.000	0.073	0.064	0.0	0.005	95.0
	α	0.00	-0.011	0.059	0.074	-1.1	1.025	100.0
	θ	1.00	1.030	0.237	0.255	3.0	0.057	95.0

Table 10: Effects of misspecified frailty distribution

Frailty	Parameter	True Value	Mean	SD	SE	Bias (%)	MSE	coverage (%)
$Gamma(\kappa, \zeta)$	β_0	1.00	1.994	0.206	0.079	99.4	1.030	0.0
$\kappa = \zeta = 1$	β_1	1.00	1.003	0.088	0.084	0.3	0.008	91.0
	β_2	1.00	1.000	0.164	0.117	-0.0	0.027	85.5
	α	1.00	0.997	0.054	0.059	-0.3	0.003	97.5
	θ	1.00	1.040	0.431	0.235	4.0	0.187	73.5
$Inv.Gaussian(\lambda, \mu)$	β_0	1.00	2.006	0.176	0.080	100.6	1.043	0.0
$\lambda = \mu = 1$	β_1	1.00	0.994	0.078	0.084	-0.6	0.006	97.0
	β_2	1.00	1.004	0.163	0.118	0.4	0.027	84.5
	α	1.00	0.994	0.054	0.059	-0.6	0.003	97.0
	θ	1.00	0.998	0.405	0.224	-0.2	0.163	72.0
$Lognormal(\mu, \sigma)$	β_0	1.00	1.991	0.184	0.080	99.1	1.016	0.0
$\mu = -\eta^2/2$	β_1	1.00	1.010	0.083	0.084	1.0	0.007	94.5
$\sigma = \eta^2$	β_2	1.00	1.005	0.160	0.118	0.5	0.025	83.5
$\eta^2 = \log(\tau^{-1} + 1)$	α	1.00	0.995	0.054	0.059	-0.5	0.003	96.5
	θ	1.00	0.949	0.461	0.215	-5.1	0.214	58.5

4.2 SIMULATION MODEL II

4.2.1 Bivariate AFT shared frailty model

The bivariate AFT shared frailty model that was considered has G independent clusters, each containing 2 sub-clusters (indexed by k). Each sub-cluster has $n/2$ observations indexed by j. The total number of observations in each dataset is $G \times n$. Data were generated based on the following log-linear AFT model:

$$\log T_{ijk} = \beta_0 + \beta_1 x_{ijk} + w_{1ik} * b_{1i} + w_{2ik} * b_{2i} + \tau * \epsilon(\alpha)_{ijk}, \tag{4.7}$$

where i = 1 . . . G independent clusters, j = 1 . . . n observations in each k sub-cluster and k =1, 2 two sub-clusters within each cluster i. w_{1ik} and w_{2ik} are $0/1$ indicator variables that specify to which sub-cluster an observation belongs. Z_{1ij} is generated from a standard normal distribution. The scale parameter τ is fixed at 1 and not estimated in simulations to simplify the problem. And the error term ϵ_{ijk} follows a family of G^{ρ} distributions where α is set at 1. The vectors b_{1i} and b_{2i} are the random effects on the sub-cluster level within a cluster i. These random effects have a joint distribution:

$$\begin{bmatrix} b_{1i} \\ b_{2i} \end{bmatrix} \sim N\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix} \begin{bmatrix} \theta_1 & \rho\sqrt{\theta_1}\sqrt{\theta_2} \\ \rho\sqrt{\theta_1}\sqrt{\theta_2} & \theta_2 \end{bmatrix} \right),$$

where θ_1 and θ_2 represent the variance components of the random effects b_{1i} and b_{2i}, respectively. When $\rho = 0$ the variance-covariance matrix Σ becomes a diagonal matrix.

Two sets of scenarios were considered in this simulation. The number of observations within a cluster is n=20. The number of clusters were G=40, 50, 60 for each setting, and 200 datasets were generated for each setting. The correlation between the frailties was set at $\rho = (0.0, 0.6)$.

4.2.2 Simulation results

Table 11 and Table 12 presents results from fitting an AFT bivariate shared frailty model (4.7) under the assumption that the two random effects (b_{1i}, b_{2i}) within a cluster i are independent $\rho = 0.0$ and dependent with a magnitude of correlation of $\rho = 0.6$, respectively. Data are generated with n=20 observations in each cluster, $\beta_0 = 1$, $\beta_1 = 1$, $\alpha = 1$, and $G = (40, 50, 60)$ for the case of 20% censoring.

The fixed effect parameters $(\beta_0, \beta_1, \alpha)$ are estimated with percent bias less than 1% when random effects are independent and less than 2% in the presence of correlation. Although there appears to be fluctuations in estimates due to sampling variability, the percent bias for fixed effect parameters decreases, in general, as the number of cluster increases. The variance components of random effects (b_{1i}, b_{2i}) are estimated with percent bias less than 2% for the independent case and less than 3% for the correlated cases. The estimates of correlation coefficients are also reasonably good with percent bias less than 2% for the independent case and less than 6% when the correlation is $\rho = 0.6$.

The asymptotic standard errors (SE) are relatively in good agreement with empirical standard errors (SD) for the fixed effect parameters β_1 and α based on the coverage probabilities. However, the asymptotic standard errors for the intercept term β_0 and the standard errors of variance of the variance components are underestimated. The empirical standard errors for all parameters are reduced by increasing the number of clusters, as expected.

Table 11: The AFT bivariate shared frailty model 1: ($\rho = 0.0$)

	Parameter	True Value	Mean	SD	SE	Bias (%)	MSE	coverage (%)
$G = 40$								
	β_0	1.00	1.006	0.147	0.136	0.6	0.021	87.0
	β_1	1.00	0.998	0.090	0.103	-0.2	0.008	94.5
	α	1.00	0.996	0.058	0.066	-0.4	0.003	93.0
	θ_1	1.00	0.981	0.332	0.295	-1.9	0.109	86.0
	θ_2	1.00	0.988	0.335	0.300	-1.2	0.111	86.5
	ρ	0.00	-0.007	0.181				
$G = 50$								
	β_0	1.00	1.006	0.144	0.119	0.6	0.021	85.5
	β_1	1.00	1.005	0.081	0.089	0.5	0.007	94.0
	α	1.00	0.997	0.055	0.058	-0.3	0.003	96.0
	θ_1	1.00	0.992	0.273	0.270	-0.8	0.074	91.0
	θ_2	1.00	1.003	0.288	0.264	0.3	0.082	88.5
	ρ	0.00	-0.012	0.176				
$G = 60$								
	β_0	1.00	0.997	0.137	0.102	-0.3	0.019	86.5
	β_1	1.00	1.001	0.076	0.081	0.1	0.006	96.0
	α	1.00	1.003	0.052	0.052	0.3	0.003	95.5
	θ_1	1.00	0.988	0.275	0.227	-1.2	0.075	87.5
	θ_2	1.00	0.983	0.274	0.233	-1.7	0.075	84.0
	ρ	0.00	-0.017	0.185				

Table 12: The AFT bivariate shared frailty model 2: ($\rho = 0.6$)

	Parameter	True Value	Mean	SD	SE	Bias (%)	MSE	coverage (%)
$G = 40$								
	β_0	1.00	1.019	0.179	0.104	1.9	0.031	76.5
	β_1	1.00	1.004	0.099	0.091	0.4	0.009	88.0
	α	1.00	0.996	0.062	0.062	-0.4	0.004	92.0
	θ_1	1.00	0.972	0.306	0.228	-2.8	0.090	78.5
	θ_2	1.00	0.962	0.273	0.221	-3.8	0.073	79.0
	ρ	0.60	0.566	0.216				
$G = 50$								
	β_0	1.00	1.013	0.156	0.091	1.3	0.024	75.5
	β_1	1.00	1.003	0.080	0.081	0.3	0.006	95.5
	α	1.00	0.996	0.062	0.055	-0.4	0.004	90.0
	θ_1	1.00	1.003	0.277	0.201	0.3	0.075	83.5
	θ_2	1.00	1.008	0.271	0.202	0.8	0.072	83.5
	ρ	0.60	0.592	0.211				
$G = 60$								
	β_0	1.00	1.010	0.138	0.082	1.0	0.019	74.0
	β_1	1.00	1.004	0.076	0.074	0.4	0.006	92.0
	α	1.00	0.998	0.052	0.050	-0.2	0.003	92.0
	θ_1	1.00	1.007	0.237	0.182	0.7	0.055	79.0
	θ_2	1.00	0.993	0.230	0.180	-0.7	0.052	81.5
	ρ	0.60	0.576	0.172				

4.2.3 Bivariate AFT nested frailty model

We consider another, yet similar, bivariate frailty model in this section.

$$\log T_{ijk} = \beta_0 + \beta_1 x_{ijk} + \eta_i + z_{1k(i)} * b_{1k(i)} + z_{2k(i)} * b_{2k(i)} + \epsilon(\alpha)_{ijk}, \tag{4.8}$$

where $i = 1 \dots G$ independent clusters, each containing 2 sub-clusters (indexed by k). Each sub-cluster has $n/2$ observations indexed by j, where $j = 1 \dots n_{ki}$ observations in each k sub-cluster. The total number of observations in each dataset is $G \times n$. The subscript notation $k(i)$ indicates that k is nested within a cluster i. More specifically, sub-cluster k is nested within a cluster η_i in model (4.8). The random effects vectors $b_{1k(i)}$ and $b_{2k(i)}$ are the random effects on the sub-cluster level within a cluster i. Since η_i is assumed to be a fixed effects covariate these random effects have the same joint distribution as in the previous model (4.7).

$$\begin{bmatrix} b_{1i} \\ b_{2i} \end{bmatrix} \sim N \left(\begin{bmatrix} 0 \\ 0 \end{bmatrix} \begin{bmatrix} \theta_1 & \rho\sqrt{\theta_1}\sqrt{\theta_2} \\ \rho\sqrt{\theta_1}\sqrt{\theta_2} & \theta_2 \end{bmatrix} \right),$$

where θ_1 and θ_2 represent the variance components of the random effects $b_{1k(i)}$ and $b_{2k(i)}$, respectively. When $\rho = 0$ the variance-covariance matrix Σ becomes a diagonal matrix. Data were generated based on above log-linear AFT nested frailty model (4.8). As before, two sets of data scenarios were considered in this simulation. The number of observations within a cluster is n=30. The number of clusters ranged between 40 and 60 (G=40, 50, 60) for each setting, and 200 datasets were generated for each setting. The correlation between the nested frailties was set at $\rho = (0.0, 0.6)$.

4.2.4 Simulation results

The results are presented in the following Table 13 and Table 14 from fitting an AFT bivariate nested frailty model (4.8) under the assumption that two random effects $(b_{1k(i)}, b_{2k(i)})$ within a cluster i are independent $\rho = 0.0$ and dependent with a magnitude of correlation of $\rho = 0.6$, respectively. Data are generated with n=30 observations in each cluster, $\beta_0 = 1$, $\beta_1 = 1$, $\alpha = 1$, and $G = (40, 50, 60)$ for the case of 20% censoring, as before.

The fixed effect parameters $(\beta_0, \beta_1, \alpha)$ are estimated with percent bias less than 2.2% when random effects are independent and less than 3% in the presence of correlation. Although there appears to be some slight fluctuations in estimation due to sampling variability, the percent bias for fixed effect parameters decreases, in general, as the number of cluster increases.

The variance components of random effects $(b_{1k(i)}, b_{2k(i)})$ are estimated with percent bias less than 4% for the independent case and less than 4.5% for the correlated cases. The estimates of the correlation coefficients have relative bias less than 2% for the independent case and less than 4.5% for the correlation of $\rho = 0.6$.

But, the asymptotic standard errors (SE) are not in a good agreement with empirical standard errors (SD) for the fixed effect parameters. Observe that asymptotic standard errors for the intercept term β_0 and also for the cluster variable η_i are seriously underestimated. For finite sample sizes, we recommend the bootstrap method for estimating standard errors. In this case, resampling is done at the level of clusters.

Table 13: The AFT bivariate nested frailty model 1: ($\rho = 0.0$)

	Parameter	True Value	Mean	SD	SE	Bias (%)	MSE	coverage (%)
$G = 40$								
	β_0	1.00	1.022	0.302	0.100	2.2	0.091	47.0
	β_1	1.00	1.005	0.073	0.082	0.5	0.005	95.0
	η	1.00	0.999	0.014	0.005	-0.1	0.000	45.0
	α	1.00	0.999	0.049	0.051	-0.1	0.002	94.5
	θ_1	1.00	0.960	0.312	0.246	-4.0	0.098	81.5
	θ_2	1.00	1.014	0.305	0.257	1.4	0.093	85.5
	ρ	0.00	-0.009	0.264				
$G = 50$								
	β_0	1.00	1.003	0.242	0.091	0.3	0.058	56.5
	β_1	1.00	0.997	0.081	0.074	-0.3	0.007	93.5
	η	1.00	1.000	0.009	0.003	0.0	0.000	56.5
	α	1.00	0.996	0.040	0.046	-0.4	0.002	97.5
	θ_1	1.00	0.975	0.265	0.220	-2.5	0.071	87.0
	θ_2	1.00	0.984	0.288	0.236	-1.6	0.083	84.0
	ρ	0.00	0.020	0.242				
$G = 60$								
	β_0	1.00	0.985	0.234	0.082	-1.5	0.054	51.0
	β_1	1.00	1.002	0.066	0.067	0.2	0.004	95.0
	η	1.00	1.000	0.008	0.003	0.0	0.000	51.0
	α	1.00	0.999	0.040	0.041	-0.1	0.002	95.0
	θ_1	1.00	1.022	0.247	0.216	2.2	0.061	85.0
	θ_2	1.00	0.981	0.249	0.203	-1.9	0.061	86.0
	ρ	0.00	-0.012	0.217				

Table 14: The AFT bivariate nested frailty model 2: ($\rho = 0.6$)

	Parameter	True Value	Mean	SD	SE	Bias (%)	MSE	coverage (%)
$G = 40$								
	β_0	1.00	1.007	0.336	0.107	0.7	0.111	45.0
	β_1	1.00	1.006	0.070	0.078	0.6	0.005	96.5
	η	1.00	1.000	0.017	0.005	-0.0	0.000	41.5
	α	1.00	0.999	0.043	0.057	-0.1	0.002	96.5
	θ_1	1.00	1.034	0.321	0.190	3.4	0.102	74.5
	θ_2	1.00	1.010	0.311	0.184	1.0	0.095	71.0
	ρ	0.60	0.596	0.176				
$G = 50$								
	β_0	1.00	0.991	0.302	0.077	-0.9	0.091	40.5
	β_1	1.00	0.993	0.071	0.070	-0.7	0.005	96.5
	η	1.00	1.001	0.012	0.003	0.1	0.000	41.0
	α	1.00	0.998	0.041	0.043	-0.2	0.002	96.0
	θ_1	1.00	1.043	0.246	0.169	4.3	0.062	85.0
	θ_2	1.00	1.030	0.325	0.166	3.0	0.106	70.5
	ρ	0.60	0.595	0.143				
$G = 60$								
	β_0	1.00	1.029	0.291	0.070	2.9	0.085	40.5
	β_1	1.00	0.998	0.068	0.064	-0.2	0.005	93.5
	η	1.00	1.000	0.009	0.002	-0.0	0.000	32.5
	α	1.00	0.997	0.036	0.039	-0.3	0.001	95.0
	θ_1	1.00	1.030	0.251	0.154	3.0	0.064	79.0
	θ_2	1.00	1.002	0.252	0.150	0.2	0.063	76.5
	ρ	0.60	0.574	0.138				

5.0 APPLICATION

5.1 NSABP PROJECT B-14: A RANDOMIZED CLINICAL TRIAL

5.1.1 Univariate AFT shared frailty model

We consider a dataset from a breast cancer clinical trial from the National Surgical Adjuvant Breast and Bowel Project (NSABP) trial, Protocol B-14. This was a phase III randomized double-blind multi-center clinical trial to determine the effectiveness of adjuvant tamoxifen therapy in patients with primary operable breast cancer who had estrogen receptor-positive tumors and no axillary lymph node involvement. More detailed description of the trial can be found in Fisher et al [16, 17, 18]. This study concluded that the treatment yielded a significantly better outcome than placebo.

Of the 2885 patients only 2817 eligible patients were randomized either to placebo and tamoxifen (1413 for placebo group and 1404 for tamoxifen group). There were 167 study sites in the study. The number of patients at each site varied from 1 to 241 with a median of 38 patients. There were twenty-seven sites with a single observation. These sites are not included in the analysis. This reduced the total number of sites to 140 and patients to 2790, with 1396 for placebo group and 1394 for tamoxifen group. The primary outcome was disease-free survival. A patient was considered to have an event in cases of recurrence, having a second primary cancer, or death (whichever occurs first). The average patient age in the analysis cohort was 55 years old ranging from 25 to 75 years old at the start of the trial.

The following univariate AFT shared frailty model was considered to examine the treatment effects and variability in log baseline survival time across the centers after adjusting for age and size of tumor:

$$\log T_{ij} = \beta_0 + \beta_1 Treatment_{ij} + \beta_2 Age_{ij} + \beta_3 Size_{ij} + b_{0i} + \epsilon(\alpha)_{ij}\,. \tag{5.1}$$

The cluster is the study site i where $i = 1...G$ with varying number of observations $j = 1...n_i$ within a cluster. The variance component of the random site effects b_{0i} is assumed to follow the normal distribution N(0, θ). The model is fitted without random effects (Model 1) and with random effects (Model 2). The error term ϵ follows the G^{ρ} family distribution with an additional parameter α.

Table 15: NSABP B-14 project

		Model 1			Model 2		
Parameters		Estimate	SE	95% CI	Estimate	SE	95% CI
Intercept	$\hat{\beta}_0$	3.095	0.086	(2.927, 3.263)	3.097	0.044	(3.012, 3.182)
Treatment	$\hat{\beta}_1$	0.469	0.074	(0.324, 0.615)	0.474	0.051	(0.374, 0.575)
Age (at 55)	$\hat{\beta}_2$	-0.005	0.004	(-0.012, 0.002)	-0.005	0.004	(-0.013, 0.004)
Tumor Size	$\hat{\beta}_3$	-0.188	0.029	(-0.246,-0.131)	-0.186	0.021	(-0.227,-0.145)
α	$\hat{\alpha}$	-0.165	0.206	(-0.568, 0.239)	-0.216	0.327	(-0.857, 0.426)
Random center effect variance	$\hat{\theta}$				0.049		

Interpretation of fixed effects parameters: The sign of the fixed effect coefficient indicates how a covariate affects the log survival times. A positive coefficient means that higher values of the covariate lead to longer log survival times. A negative coefficient means that higher values of the covariate lead to shorter log survival time. In terms of the magnitude, the coefficients can be interpreted either as the time ratio or percentage change in survival time. Time ratio is calculated by exponentiating the coefficient: $\exp(\hat{\beta_k}\delta)$ where δ is amount of change in the covariate value. Percentage change is calculated by $100[\exp(\hat{\beta_k}\delta) - 1]$. See Appendix **??** for the details.

Based on the results from the Model 2, the sign of the coefficient of treatment effect is positive indicating treatment increases the survival time compared to that of placebo group, as expected, by 61%. The age variable was centered at its mean in the analysis. As the age of the patient increases from 55 years old the log survival time decreases. A one year increase in age results in a 0.5% decrease in the survival time. As the size of tumor increases the log survival time decreases. A one centimeter increase in tumor size results in a 17% decrease in the survival time.

Variance component of random site effects: The magnitude of the variance of the site effect is estimated to be 0.049 (Model 2 in the Table 15). This indicates that there appears to be very small variability in the baseline log survival time across the sites. Figure 5.1 shows a histogram with smoothed density curves (dotted line) and normal density overlay (solid line) of the predicted random effects. The predicted frailties appear to be centered around zero and symmetrically distributed.

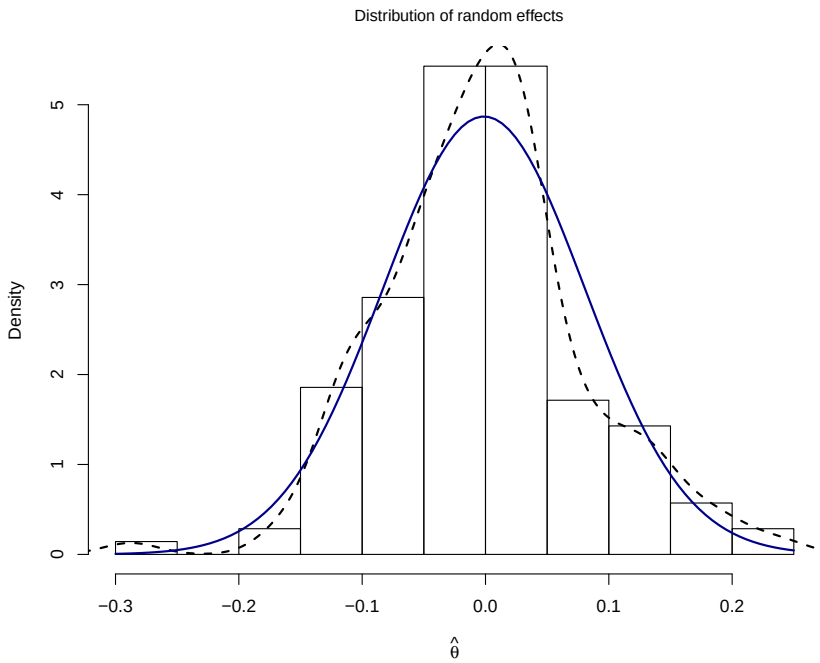

Figure 5.1: Distribution of predicted random effects

Figure 5.2 shows the non-parametric and parametric hazard estimates for the NSABP B-14 study. The dotted lines are the non-parametric empirical life-table estimates. The upper line is for the placebo group and the lower line is for the treatment group. The solid lines in Figure 5.2 represent the corresponding parametric hazard estimates from the Model 2. The tests for the proportional hazard assumption yielded p-value < 0.001. In general, the parametric estimates depicts the non-parametric hazard rates well. Of note, the large early treatment difference in the first two years shown by the non-parametric estimates is reduced by the parametric estimates. This is partly due to the fact that our model takes into account age and tumor size.

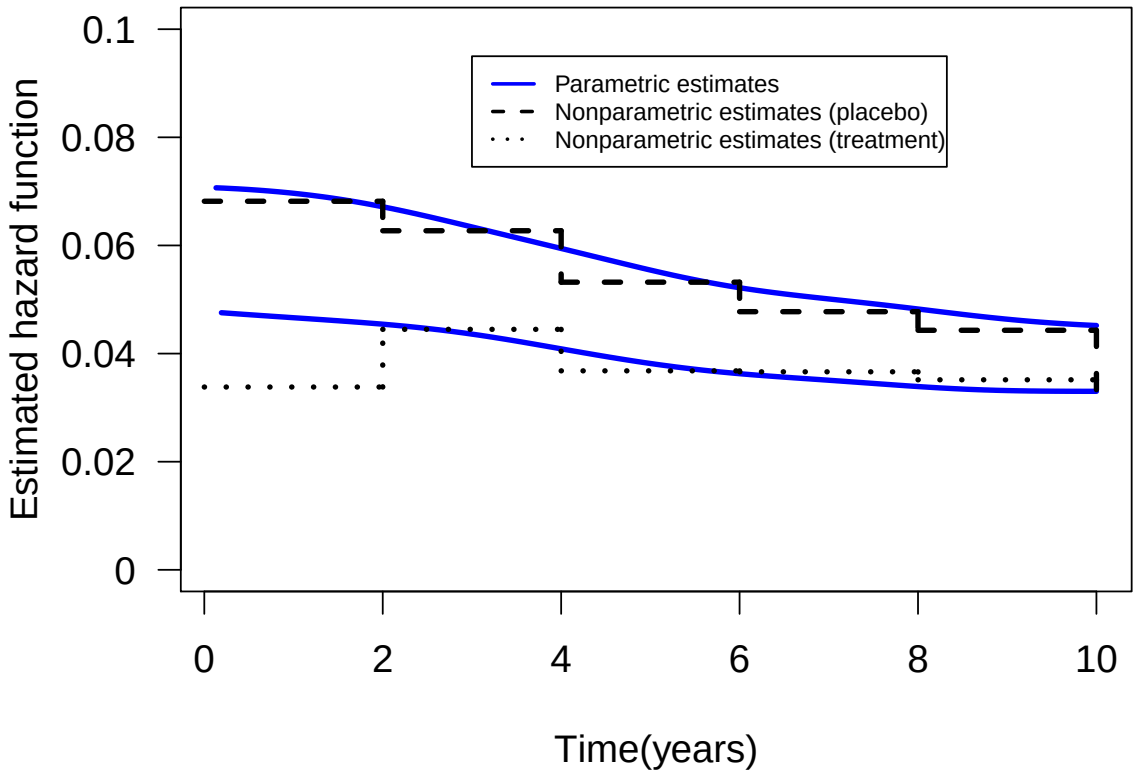

Figure 5.2: Non-parametric and parametric hazard estimates for NSABP B-14 data

5.1.2 Bivariate AFT nested frailty model

In this section, we considered each age group as a unit of cluster. In other words, the age group is considered as a cluster instead of the study site. Further, we categorized patients into two groups based on the size of tumor (size $\leq 2.0cm$ and size $> 2.0cm$) within each age group. Therefore, two tumor size groups $k = 1, 2$ which are nested within the same age group i are considered as sub-clusters and thus the levels of random effects. This was done to examine variability in the baseline log survival times across the age groups with the size of tumor. With NSABP B-14 data, forty age groups were formed from age 31 to age 70. The number of patients in each age group varied from 10 to 124 with a median of 73 patients. There are n=1381 patients for placebo group and n=1386 patients for treatment group. The following AFT model with nested bivariate random effects is considered for the data:

$$\log T_{ijk} = \beta_0 + \beta_1 Treatment_{ijk} + Age_i + size_{small.k(i)} * b_{1k(i)} + size_{large.k(i)} * b_{2k(i)} + \epsilon(\alpha)_{ijk}, \qquad (5.2)$$

where i = 1 ... G independent age clusters, $j = 1 \ldots n_{jk}$ observations in each k sub-cluster and k =1, 2 two sub-clusters within each cluster i. The subscript notation $k(i)$ indicates k is nested with a cluster i. Model (5.2) was run with 2000 EM iterations. In addition, B=200 bootstrap resampling with replacement was done to estimate standard errors for this model. Results are summarized in Table 16.

Interpretation of fixed effects parameters: The signs of the fixed effect coefficients for treatment and age are consistent with the previous two models (Model 1 and Model 2 from the Table 15). The magnitude of treatment effect is slightly reduced from the univariate random effects model such that treatment increases the survival time by 57%. It is possible that the magnitude of treatment effects in Model 3 is slightly decreased by modeling the tumor size as nested random effects. The magnitude of the age effect did not change. A one year increase in age resulted in a 0.5% decrease in

the survival time. There was noticeable change in the estimate of α. It decreased from $\hat{\alpha} = -0.216$ in Model 2 to $\hat{\alpha} = -0.380$ in Model 3. This may indicate that the early treatment differences were slightly greater in the Model 3 compared to the Model 2.

Table 16: NSABP B-14 project: Bivariate AFT nested frailty model

		Model 3			
Parameters		Estimate	SE_{Louis}	SE_{Boot}	95% CI_{Boot}
Intercept	$\hat{\beta}_0$	2.848	0.041	0.108	(2.635,3.061)
Treatment	$\hat{\beta}_1$	0.450	0.038	0.069	(0.315,0.585)
Age_i	$\hat{\eta}_i$	-0.005	0.001	0.004	(-0.013,0.003)
α	$\hat{\alpha}$	-0.380	0.346	0.294	(-0.955,0.195)
$Age(Size \leq 2.0cm)_{1k(i)}$	$\hat{\theta}_1$	0.052	0.015	0.021	(0.011,0.093)
$Age(Size > 2.0cm)_{2k(i)}$	$\hat{\theta}_2$	0.098	0.025	0.038	(0.023,0.173)
$correlation(\theta_1, \theta_2)$	$\hat{\rho}$	0.028		0.089	(-0.147,0.203)

Variance components of random size effects: Though the magnitudes of the variance components of tumor size effect are not exceedingly large, patients with larger tumor sizes ($> 2.0cm$) have twice the variability in survival time across the age groups as patients with smaller tumor sizes ($\leq 2.0cm$). There is a slight positive correlation between these two tumor size sub-clusters, but with a small magnitude ($\hat{\rho} = 0.028$). This positive correlation can be interpreted as the random effects of two tumor size sub-clusters moving together in the same direction. Figure 5.3 depicts the predicted random effects by the tumor size sub-clusters. In Figure 5.3 (a), the predicted random effects of patients with tumor size $\leq 2.0cm$ are shown. The random effects are mostly close to zero or above the zero-line across age groups, indicating positive outlook on the survival time. However, in Figure 5.3 (b), the predicted random effects of patients with tumor size $> 2.0cm$ depicts more variation across the age groups and they also vary around the zero line but are mostly under the zero line, indicating its

association with decreased survival time across the age groups. Especially, patients with larger tumor size have lower survival time across almost all the age groups above 60 years old.

When marginal Akaike's information criterion (AIC) is compared across the models Model 3 does not necessarily outperform the others. In fact, Model 2 appears to be the best choice (Table 17).

Table 17: NSABP B-14 data: Values for $-2\sum \log \hat{L}$ and mAIC

	Model 1	Model 2	Model 3
-2$\sum \log \hat{L}$	8982.40	8918.96	8986.12
s	0	1	3
p	5	5	4
$mAIC$	8992.4	8930.96	9000.12

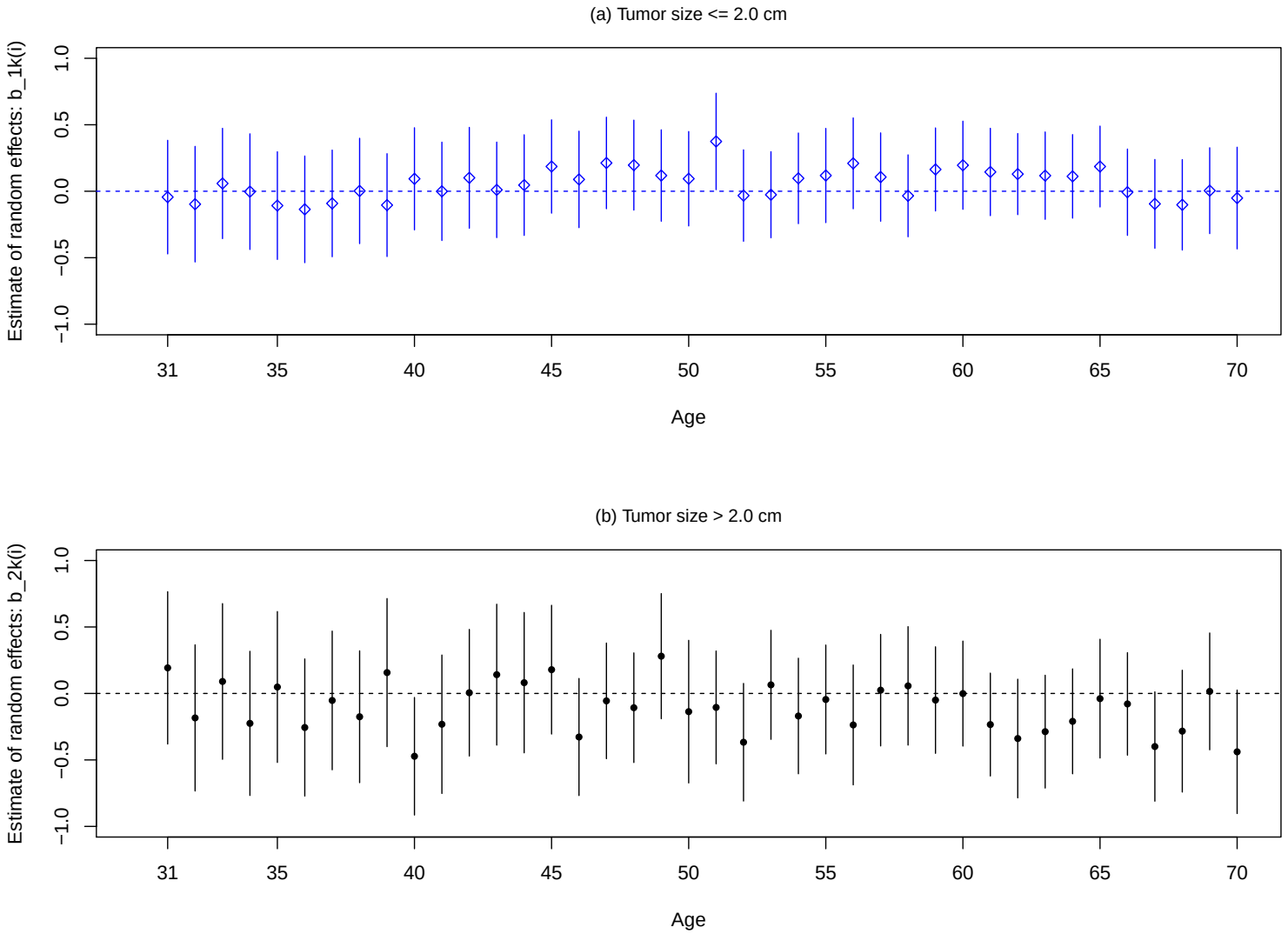

Figure 5.3: Predicted random effects by age and tumor size: NSABP B-14 data:
The predicted random effects for tumor size sub-cluster k nested with each age cluster i, that is $\hat{b_{k(i)}} = E[b_{k(i)}|y, \hat{\theta}]$ for $i = 1, \ldots, G$ and $k = 1, 2$, and the corresponding 95 per cent credibility intervals using the normal approximation are plotted. Predicted random effects are based on the results after fitting a bivariate AFT nested model as in the equation (5.2)

6.0 DISCUSSION

We proposed a Stochastic Expectation-Maximization (StEM) estimation procedure for AFT models with random effects to model non-proportional attenuating hazards for the correlated survival times. The stochastic EM method [8, 13, 14] is conceptually simple in that the computational complexity of the likelihood is avoided by imputing the latent data so that the complete data likelihood can be used. This estimation method is not restricted to any specific distribution of data and can be used for a broad range of statistical models. The mean of the stationary distribution is considered to be an estimate of the parameter of interest and it approaches the maximum likelihood estimator in the order of $O(1/n)$ as the sample size n increases. In addition, results of consistency and asymptotic normality of this mean estimator have been established previously for specific examples (Diebolt and Celeus, 1993) [13], discussed (Ip, 1994) [33] and generalized (Nielsen, 2000) [55].

The NSABP B-14 data are characterized by attenuating hazards (Figure 5.2). It is interesting to note that the hazards for the control group continually decrease over time which results in the attenuating pattern of the treatment effect. The reason for the continual decrease in the hazards for the control group is not clear. However, it is plausible to conjecture that there may have been a survivorship effect. Those who survived without receiving the treatment for the long term are more likely to be strong survivors and more robust against the event. Their risk of experiencing an event becomes more comparable with those who survived with the treatment as time goes by.

The NSABP B-14 study was a double-blind study. Patients were treated either with tamoxifen (10 mg twice a day, given orally) or with placebo, similarly administered. Placebo and tamoxifen tablets were indistinguishable on the basis of taste and physical appearance. The pharmacologic formulation of the placebo was identical to that of the tamoxifen except for the absence of active drug. Double blinding was used so that neither medical personnel nor the patients could determine the type of treatment administered. Both groups received the treatment for five years. Compliance with protocol-designed therapy during the first five years of the study was similar for both groups [16, 17, 18]. It is conceivable that hazard attenuation may reflect a placebo effect, though this is usually known as a short term effect. The important question is whether the treatment differences in the presence of attenuation are still significant. The answer was yes using our proposed model (Table 15 and Table 16). There was a slight increase in the hazard in the treatment group between year 2 and year 4 (Figure 5.2). This implies a higher incidence of events for the treatment group during this period but that may have been caused by a random fluctuation. However, this increase may be worth further investigation.

Another finding was that the large early treatment differences observed with the unadjusted non-parametric life-table method was reduced by the adjusted analysis. This finding was interesting in that the large initial treatment difference was reduced once analysis accounted for the differences in age and tumor size of the patients.

By adapting a family of G^{ρ} distribution for the errors, one can model such non-proportional attenuating hazards in the NSABP B-14 clinical trial. Thus, this work expanded the perimeter of error distributions in the accelerated failure time literature. For the frailty distribution, the multivariate normal distribution was used. Though computationally intensive, the multivariate normal distribution enables us to model either positive or negative correlation among the frailties. These models together provide a flexible approach to the parametric analysis of correlated survival data.

The Stochastic Expectation-Maximization (StEM) is a stochastic variant method of EM algorithm, for which that additional steps are needed to find the variance-covariance matrix. Computing the variance estimates is a major issue using EM. When the EM algorithm is used, Louis (1982) derived a procedure for obtaining the asymptotic variance-covariance matrix. Louis (1982) showed that the observed information is the difference of complete and missing data information. Since the EM algorithm deals with data composed of complete data and latent missing data, Louis' method has been conceptually intuitive and thus appealing to the EM algorithm users. Various authors have come up with different ways to estimate asymptotic standard errors including the methods proposed by Meng and Rubin (1991), Meilijson (1989) and Carlin (1987) [54, 7, 53]. However, Efron (1992) [15] pointed out that these methods are basically delta methods and they often underestimate the variance. Thus, the Louis method [50] was used to estimate the asymptotic standard errors in this dissertation. It works well for the fixed effects parameters with continuous variables in a simple setting. However, overall, the finite sample properties are not optimal with the sample sizes studied in this dissertation. It does not have a good coverage rate for the intercept parameter and also underestimates standard errors for binary variables. Underestimation is greater in a more complicated model setting. Binary variables are important since one is often interested in treatment effects which are commonly coded as a binary variable. One possible explanation for underestimation in our case might be due to the approximation to the observed information. This is because of the stochastic nature of StEM algorithm, i.e., imputing the latent data. This point was also noted by Diebolt and Ip (1994) [14].

Further research is needed in the area of estimation of asymptotic variance-covariance matrix after StEM algorithm. In general, we recommend a non-parametric bootstrap method for standard error estimation. It is a non-parametric alternative to an analytical solution. It is conceptually simple and can be carried out by implementing the resampling of the observed dataset and repeat the estimation.

This can be implemented in the programming, though the resampling and re-estimation will require more time on top of already very time consuming simulations using the StEM algorithm.

In this dissertation, simulation studies showed that fixed effects parameters and variance components were estimated with little percentage bias as the sample size increased. However, increasing the number of clusters at a fixed cluster size had only a small effect on reducing the bias in the estimates of variance components of the random effects. Adequate cluster size is also needed for acceptable bias in estimates of the variance components in these settings.

These simulations were carried out for the cases with equal sample size across the clusters. It is expected that the fixed effects parameters will be less affected by the unequal cluster sizes than the random effects parameters as long as the overall sample size is large enough. However, it is expected that the unequal cluster sizes will impact on the estimates of the random effects, and thus the variance components. This will be reflected in the credibility intervals for the estimated random effects. The intervals would be wider if the corresponding cluster sizes are smaller. Then, caution is required when interpreting these results. More weights should be given for those clusters with large cluster sizes. Further simulation studies are needed to quantify the exact effects of varying cluster sizes.

As noted by various authors [36, 44] the AFT model exhibits robustness of parameter estimation in the presence of misspecification and presence of heterogeneity in the study population. This is one of the main advantages of the AFT model over the Cox proportional hazards model [31]. In our simulation study, misspecifying several frailty distributions did not affect parameter estimation, except for the intercept term. In practice, clinicians are often interested in the effect of treatment. When the question of interest is centered around the treatment effect only, the bias in the intercept does not introduce a major issue. However, when one wants to evaluate the average baseline log-survival time the over- or under-estimation of intercept will introduce a bias. As to why the bias is

occurred in the intercept term in the misspecified frailty model, that needs further investigation.

Our simulation study also showed that the estimates of the intercept and variance components are biased when the error distribution is assumed to be a log-normal distribution. However, other fixed effects parameters were estimated well. Although parametric MLEs are consistent and most efficient when the baseline hazard is correctly specified, they are generally biased when the baseline hazard is misspecified. Strong distributional assumption of the survival times is a drawback of parametric AFT models. This can be relaxed in the semi-parametric AFT model setting and is an area for the future research. Other areas of further investigation include an AFT model-based sample size calculation, model diagnostics and model comparisons using conditional Akaike information criterion (cAIC).

APPENDIX A

TIME RATIO AND PERCENTAGE CHANGE

Consider a log-linear AFT model from the equation (2.8) in the section 2.2:

$$\log T = \beta^T Z + b^T W + \tau * \epsilon, \tag{A.1}$$

where T is the failure time, Z and W are covariate vectors for the fixed and random effects, respectively, β is the vector of fixed effects, and b is the vector of random effects. Then, the marginal effect of Z_k on $\log T$ is

$$\frac{\partial \log T}{\partial T} = \beta_k \tag{A.2}$$

And from the equation (A.1)

$$T = \exp(\beta^T Z) \exp(b^T W) \exp(\tau * \epsilon). \tag{A.3}$$

Suppose we change the value of some covariate Z_k, by some amount δ, the ratio of the survival times is

$$\begin{aligned} \frac{T(Z_k + \delta)}{T(Z_k)} &= \exp[(Z_k + \delta) - Z_k]\hat{\beta}_k \\ &= \exp(\hat{\beta}_k \delta). \end{aligned} \tag{A.4}$$

If δ is just one unit, this simplifies to

$$\frac{T(Z_k + \delta)}{T(Z_k)} = \exp \hat{\beta}_k, \tag{A.5}$$

where $\exp \hat{\beta}_k$ is known as the time ratio. We can interpret this in a similar way to the hazard ratio in a Cox proportional hazards model. For example, if the exponentiated coefficient, $\exp \hat{\beta}_k = 1.50$, then we say that a one unit increase in Z_k increases the survival time by a factor of 1.5. In other words, the survival time is 1.5 times longer.

Another interpretation is to use the percentage change in the survival time associated with a change in the value of some covariate, Z_k, by some amount δ:

$$\text{Percentage change} = 100[\exp(\hat{\beta}_k \delta) - 1]. \tag{A.6}$$

If δ is just one unit and the exponentiated coefficient, $\exp \hat{\beta}_k = 1.50$, then we say that a one unit increase in Z_k increases the surival time by 50%.

(References: Jenkins, Stephen P. 2008. "Survival Analysis." Unpublished manuscript, Institute for Social and Economic Research, University of Esses, Colchester.)

APPENDIX B

R PROGRAM FOR AFT MODEL WITH NESTED RANDOM EFFECTS

```
1  ##############################################################################
2  # nested_AFT_model_with_random_effects.R
3  # simulation program
4  ##############################################################################
5  #remove(list=ls(all=TRUE))
6  library(lattice)
7  library(mcmc)
8  library(coda)
9  library(lattice)
10 library(MASS)
11 library(MCMCpack)
12 library(mvtnorm)
13 library(SamplerCompare)
14 library(splines)
15 library(survival)
16 library(smoothSurv)
17 library(bayesSurv)
18 library(xtable)
19 library(boot)
20
21 #-----------------------------------------------------------------------------
22 # Set the directory for output files
23 #-----------------------------------------------------------------------------
24 directory  <-"c:\\_dissertation\\outputs\\BivariateModel\\optim\\nested\\corr06\\G60n30\\" # need to change
25
26 #-----------------------------------------------------------------------------
27 # Generate data -- Simulation parameters
28 #-----------------------------------------------------------------------------
29   NUM.DATA <- 200
30   simul.summary <- matrix(NA,nrow=NUM.DATA,ncol=26)
31   save.seeds    <- matrix(NA,nrow=NUM.DATA,ncol=1)
32   begin.time <- date()
33
34   n.cluster  <- 60   # CHANGE HERE  overall number of clusters
35   n.obs      <- 30   # CHANGE HERE  n.obs/2 per each sub-cluster
36   cluster    <- c(rep(1:n.cluster,each=n.obs))
37   n.sub.cluster <- 2
38   EMsteps   <- 250
39   initial   <- 50  # first 50 estimates will not be included to estimate parameter means
40   ARMsteps  <- 1
41   n.Gibbs   <- 1   ## MCEM if >1 otherwise SEM
42   T.alpha   <- 1
43   my.theta1 <- 1
44   my.corr   <- 0.6
45   my.sigma  <- matrix(c(1,my.corr,my.corr,1),2,2)
46   T.eta.grp <- 1
47   true.value<- c(1,1,T.eta.grp,T.alpha,my.sigma[1,1], my.sigma[2,2], my.sigma[1,2], my.sigma[1,2], my.sigma
         [1,2])
48
49 #-----------------------------------------------------------------------------
50 # Functions
51 #-----------------------------------------------------------------------------
52 #Run a function that creates a diagonal matrix
53 #source http://tolstoy.newcastle.edu.au/R/help/04/05/1322.html
54 bdiag <- function(x){
55      if(!is.list(x)) stop("x not a list")
56      n <- length(x)
57      if(n==0) return(NULL)
58      x <- lapply(x, function(y) if(length(y)) as.matrix(y) else
59 stop("Zero-length component in x"))
60      d <- array(unlist(lapply(x, dim)), c(2, n))
61      rr <- d[1,]
62      cc <- d[2,]
```

```
     rsum <- sum(rr)
     csum <- sum(cc)
     out <- array(0, c(rsum, csum))
     ind <- array(0, c(4, n))
     rcum <- cumsum(rr)
     ccum <- cumsum(cc)
     ind[1,-1] <- rcum[-n]
     ind[2,] <- rcum
     ind[3,-1] <- ccum[-n]
     ind[4,] <- ccum
     imat <- array(1:(rsum * csum), c(rsum, csum))
     iuse <- apply(ind, 2, function(y, imat) imat[(y[1]+1):y[2],
(y[3]+1):y[4]], imat=imat)
     iuse <- as.vector(unlist(iuse))
     out[iuse] <- unlist(x)
     return(out)
 }
#-------------------------------------------------------------------------
# Function of b1 given random.b2
#-------------------------------------------------------------------------
 posterior.b.i.z.all.b1 <- function(random.b1){
    new.data <- my.data[which(cluster==i),]  #  by site
    new.b1 <- random.b1
    new.b2 <- random.b2
    all.b  <- matrix( c(random.b1, random.b2), 1, n.sub.cluster) # so 1x2 matrix for b rand.effects, n.sub.
         cluster is always 2 for now
  Z <- cbind(1,as.matrix(subset(new.data,select= c(x1,eta.grp))))       # data for fixed effects  200x3
  newW <- as.matrix(mymat2[[i]])

  xbeta <- c(beta1,beta2, beta3)
  W.b     <-  newW %*% t(all.b) # i runs from 1 .. to n.cluster , change 7/10/2011
  Z.beta  <- Z %*% xbeta    # 20x3 times 3x1 generates 20x1 Z.beta matrix
  logT    <- log(new.data$t)
  T       <- new.data$t
  delta   <- as.matrix(new.data$status, nrow(new.data),ncol=1)
  summyb <- sum(apply(all.b,1, function(x) { #print(x)
                 myb <- matrix(x,1,2,byrow=TRUE)
                 myb %*% solve(sigma.mat) %*% t(myb)}))
  logl.1 <- (-1)*sum(delta*(Z.beta+W.b)+(delta+exp(-alpha))*log(1+exp(alpha+logT-Z.beta-W.b)))
  logl.2 <- (-n.cluster*n.sub.cluster/2)*(log(2*pi)) + (n.cluster/2)*log(det(solve(sigma.mat)))- summyb/2
  logl.b <- logl.1 + logl.2
  return(logl.b)
   }
#-------------------------------------------------------------------------
# Function of b2 given random.b1
#-------------------------------------------------------------------------
 posterior.b.i.z.all.b2 <- function(random.b2){
    new.data <- my.data[which(cluster==i),]  #  by site
    new.b1 <- random.b1
    new.b2 <- random.b2
    all.b  <- matrix( c(random.b1, random.b2), 1, n.sub.cluster) # so 1x2 matrix for b rand.effects
  Z <- cbind(1,as.matrix(subset(new.data,select= c(x1, eta.grp))))      # data for fixed effects  200x3
  newW <- as.matrix(mymat2[[i]])
  xbeta <- c(beta1,beta2,beta3)
  W.b     <-  newW %*% t(all.b) # i runs from 1 .. to n.cluster , change 7/10/2011
  Z.beta  <- Z %*% xbeta    # 20x3 times 3x1 generates 20x1 Z.beta matrix
  logT    <- log(new.data$t)
  T       <- new.data$t
  delta   <- as.matrix(new.data$status, nrow(new.data),ncol=1)
  summyb <- sum(apply(all.b,1, function(x) { #print(x)
                 myb <- matrix(x,1,2,byrow=TRUE)
                 myb %*% solve(sigma.mat) %*% t(myb)}))
  logl.1 <- (-1)*sum(delta*(Z.beta+W.b)+(delta+exp(-alpha))*log(1+exp(alpha+logT-Z.beta-W.b)))
  logl.2 <- (-n.cluster*n.sub.cluster/2)*(log(2*pi)) + (n.cluster/2)*log(det(solve(sigma.mat)))- summyb/2
  logl.b <- logl.1 + logl.2
  return(logl.b)
 }
#-------------------------------------------------------------------------
#  calculate E[G_{ij}|Y, \theta^(k)] function with given expectations from E-step
#-------------------------------------------------------------------------
  E.gij <- function(beta1,beta2,beta3,alpha,old.beta, old.alpha, bb){
    new.data  <- my.data
    e.beta <- c(beta1, beta2, beta3)
    newb    <- bb  # this should be "stacked" b.vector
  W.b    <- W %*% newb
  Z.beta <- Z %*% old.beta
  logT       <- log(new.data$t)
  T          <- new.data$t
  delta      <- as.matrix(new.data$status, nrow(new.data),ncol=1)
  g.log <- log(1+exp(alpha+(logT-Z.beta-W.b))) # component 1
    e.alpha <- alpha # expected value of alpha at (k)th iteration
  W.b    <- W %*% c(newb)     # 200x10 time 10x1 genereate 20x1 W.b matrix
  Z.beta <- Z %*% old.beta    # 200x3 times 3x1 generates 20x1 Z.beta matrix
  logT       <- log(new.data$t)
  T          <- new.data$t
  delta      <- as.matrix(new.data$status, nrow(new.data),ncol=1)

  dg.dbeta.lower <- exp(-(alpha+(logT-Z.beta-W.b)))+1
  dg.dbeta1   <- Z[,1]/dg.dbeta.lower
```

```
  dg.dbeta2   <- Z[,2]/dg.dbeta.lower
  dg.dbeta3   <- Z[,3]/dg.dbeta.lower
  xdg.dbeta  <- cbind(dg.dbeta1,dg.dbeta2, dg.dbeta3)  # matrix
  dg.dbeta   <- xdg.dbeta %*% (e.beta - old.beta)  # component 2
  dg.alpha.lower <-exp(-(alpha+(logT-Z.beta-W.b)))+1
  xdg.alpha       <- 1/dg.alpha.lower
  dg.alpha       <- xdg.alpha %*% (e.alpha - old.alpha) # component 3
  out.gij <-  g.log + dg.dbeta + dg.alpha
  return(out.gij)
  }
#-------------------------------------------------------------------------------
#  calculate Q1
#-------------------------------------------------------------------------------
  Q1 <- function(beta1, beta2, beta3, alpha, xEgij) {
    new.data  <- my.data
    beta <- c(beta1, beta2,beta3)
    #print(paste("printing beta -->", beta))
  Z.beta     <- Z %*% beta    # 200x3 times 3x1 generates 200x1 Z.beta matrix
  logT       <- log(new.data$t)
  delta      <- as.matrix(new.data$status, nrow(new.data),ncol=1)
  out.Q1 <-  (-1)*sum(delta*Z.beta + (delta+exp(-alpha))*xEgij)
  # E.g = E[g_{ij}|Y, \theta^{k}] is from above
  return(out.Q1)
   }
#-------------------------------------------------------------------------------
#  calculate Q2  -- updated for bivariate case
#-------------------------------------------------------------------------------
  Q2 <- function(bb, sigma) {
    #G is number of cluster
    G <- n.cluster
    d <- n.sub.cluster # number of random effects
  allmyb <- do.call("rbind",b.vector.test)
  summyb <- sum(apply(allmyb,1, function(x) { #print(x)
                 myb <- matrix(x,1,2,byrow=TRUE)
                 myb %*% solve(sigma.mat) %*% t(myb)}))
    out.Q2 <-    (-0.5)*G*d*log(2*pi) - (G/2)*log(det(sigma.mat))- summyb/2
    return(out.Q2)
  }
#-------------------------------------------------------------------------------
#
#-------------------------------------------------------------------------------
 new.Q1 <- function(theta, bb){
    new.data  <- my.data
    # these are current estimates at k-th iteration
    old.theta <- c(get.beta1[k,],get.beta2[k,],get.beta3[k,],get.alpha[k,]) # c(beta1,beta2,beta3,alpha)
    newb    <- bb  # this should be "stacked" b.vector [b1 b2]
  W.b    <- W %*% newb       # 200x10 time 10x1 genereate 20x1 W.b matrix
  Z.beta <- Z %*% old.theta[1:3]    # 200x3 times 3x1 generates 20x1 Z.beta matrix
  logT       <- log(new.data$t)
  T          <- new.data$t
  delta      <- as.matrix(new.data$status, nrow(new.data),ncol=1)
  # This is g_{ij}^{k}
  g.log <- log(1+exp(old.theta[4]+(logT-Z.beta-W.b))) # component 1
  #e.alpha <- alpha # expected value of alpha at (k)th iteration
  e.alpha <- theta[4]

  dg.dbeta.lower <- exp(-(old.theta[4]+(logT-Z.beta-W.b)))+1
  dg.dbeta1<- Z[,1]/dg.dbeta.lower
  dg.dbeta2<- Z[,2]/dg.dbeta.lower
  dg.dbeta3<- Z[,3]/dg.dbeta.lower
  xdg.dbeta<- cbind(dg.dbeta1,dg.dbeta2,dg.dbeta3)  # matrix
  dg.dbeta <-  xdg.dbeta %*% (theta[1:3] - old.theta[1:3])  # component 2
  dg.alpha.lower <-exp(-(old.theta[4]+(logT-Z.beta-W.b)))+1
  xdg.alpha<- 1/dg.alpha.lower
  dg.alpha<- xdg.alpha %*% (theta[4] - old.theta[4]) # component 3
  new.gij <- g.log + xdg.dbeta %*% (theta[1:3] - old.theta[1:3]) + xdg.alpha %*% (theta[4] - old.theta[4])
  my.Q1<- (-1)*(-1)*sum(delta*(Z %*% theta[1:3]) + W.b + (delta+exp(-theta[4]))*(g.log + xdg.dbeta %*% (theta
       [1:3] - old.theta[1:3]) + xdg.alpha %*% (theta[4] - old.theta[4])) )
  my.el<- (-1)*(-1)*sum(delta*(Z %*% theta[1:3]) + delta*W.b + (delta+exp(-theta[4]))*(log(1+exp(theta[4]+logT-
       Z %*% theta[1:3]-W.b))))
  return(my.el)
   } # end of  new.Q1 function
#-------------------------------------------------------------------------------
#
#-------------------------------------------------------------------------------
 calc.Q1 <- function(theta, bb){
    new.data  <- my.data
    # these are current estimates at k-th iteration
    old.theta <- c(get.beta1[k,],get.beta2[k,],get.beta3[k,],get.alpha[k,]) # c(beta1,beta2,alpha)
    newb    <- bb  # this should be "stacked" b.vector [b1 b2]
  W.b    <- W %*% newb       # 200x10 time 10x1 genereate 20x1 W.b matrix
  Z.beta <- Z %*% old.theta[1:3]    # 200x3 times 3x1 generates 20x1 Z.beta matrix
  logT       <- log(new.data$t)
  T          <- new.data$t
  delta      <- as.matrix(new.data$status, nrow(new.data),ncol=1)
  # This is g_{ij}^{k}
  g.log <- log(1+exp(old.theta[4]+(logT-Z.beta-W.b))) # component 1
  e.alpha <- theta[4]
  dg.dbeta.lower <- exp(-(old.theta[4]+(logT-Z.beta-W.b)))+1
```

```
  dg.dbeta1<- Z[,1]/dg.dbeta.lower
  dg.dbeta2<- Z[,2]/dg.dbeta.lower
  dg.dbeta3<- Z[,3]/dg.dbeta.lower
  xdg.dbeta<- cbind(dg.dbeta1,dg.dbeta2,dg.dbeta3)  # matrix
  dg.dbeta <-  xdg.dbeta %*% (theta[1:3] - old.theta[1:3])  # component 2
  dg.alpha.lower <-exp(-(old.theta[4]+(logT-Z.beta-W.b)))+1
  xdg.alpha<- 1/dg.alpha.lower
  dg.alpha <- xdg.alpha %*% (theta[4] - old.theta[4]) # component 3
  new.gij <- g.log + xdg.dbeta %*% (theta[1:3] - old.theta[1:3]) + xdg.alpha %*% (theta[4] - old.theta[4])
  my.Q1 <- (-1)*sum(delta*(Z %*% theta[1:3]) + W.b + (delta+exp(-theta[4]))*(g.log + xdg.dbeta %*% (theta[1:3]
       - old.theta[1:3]) + xdg.alpha %*% (theta[4] - old.theta[4])) )
  return(my.Q1)
   } # end of  new.Q1 function
#-------------------------------------------------------------------------------
#
#-------------------------------------------------------------------------------
 calc.el <- function(theta, bb){
    new.data  <- my.data
    # these are current estimates at k-th iteration
    old.theta <- c(get.beta1[k,],get.beta2[k,],get.beta3[k,],get.alpha[k,]) # c(beta1,beta2,alpha)
    newb <- bb  # this should be "stacked" b.vector [b1 b2]
  W.b  <- W %*% newb       # 200x10 time 10x1 genereate 20x1 W.b matrix
  Z.beta <- Z %*% old.theta[1:3]    # 200x3 times 3x1 generates 20x1 Z.beta matrix
  logT <- log(new.data$t)
  T    <- new.data$t
  delta<- as.matrix(new.data$status, nrow(new.data),ncol=1)
  # This is g_{ij}^{k}
  g.log <- log(1+exp(old.theta[4]+(logT-Z.beta-W.b))) # component 1
  #e.alpha <- alpha # expected value of alpha at (k)th iteration
  e.alpha <- theta[4]
  dg.dbeta.lower <- exp(-(old.theta[4]+(logT-Z.beta-W.b)))+1
  dg.dbeta1<- Z[,1]/dg.dbeta.lower
  dg.dbeta2<- Z[,2]/dg.dbeta.lower
  dg.dbeta3<- Z[,3]/dg.dbeta.lower
  xdg.dbeta<- cbind(dg.dbeta1,dg.dbeta2,dg.dbeta3)  # matrix
  dg.dbeta <-  xdg.dbeta %*% (theta[1:3] - old.theta[1:3])  # component 2
  dg.alpha.lower <-exp(-(old.theta[4]+(logT-Z.beta-W.b)))+1
  xdg.alpha<- 1/dg.alpha.lower
  dg.alpha <- xdg.alpha %*% (theta[4] - old.theta[4]) # component 3
  new.gij  <- g.log + xdg.dbeta %*% (theta[1:3] - old.theta[1:3]) + xdg.alpha %*% (theta[4] - old.theta[4])
  my.el <- (-1)*sum(delta*(Z %*% theta[1:3]) + delta*W.b + (delta+exp(-theta[4]))*(log(1+exp(theta[4]+logT-Z %*
       % theta[1:3]-W.b))))
  return(my.el)
   } # end of  calc.el function

#-------------------------------------------------------------------------------
#   END OF ALL FUNCTIONS
#-------------------------------------------------------------------------------

  for (kk in (1:NUM.DATA)) {

#-------------------------------------------------------------------------------
# Generate data -- based on nested AFT model with bivariate random effects
#-------------------------------------------------------------------------------
my.seed <- round(100000*runif(1))
save.seeds[kk,] <- my.seed
set.seed(my.seed)   # set the seed for random number generator

x1 <- c(rnorm(n.cluster*(n.obs), mean=0, sd=1))
z1 <- rep(c(rep(1,n.obs/2),rep(0,n.obs/2)), n.cluster)
z2 <- rep(c(rep(0,n.obs/2),rep(1,n.obs/2)),n.cluster)
U  <- runif(n.cluster*(n.obs)) # uniform random variable
e3 <- (-T.alpha) + log(exp((-1)*(exp(T.alpha))*(log(1-U))))-1)
my.b1.b2 <- mvrnorm(n=n.cluster, mu=c(0,0), Sigma=my.sigma) # bivariate data generateion from library(MASS)
xmyb1  <- rep(my.b1.b2[,1], each=n.obs)
xmyb2  <- rep(my.b1.b2[,2], each=n.obs)
eta.grp <- cluster
tt2 <- exp(1+x1+eta.grp+z1*xmyb1+z2*xmyb2+e3)
cc  <- c(rexp(n.cluster*(n.obs), 1/quantile(tt2, 0.815)))
status <- ifelse(cc<=tt2, 0,1)
t   <- pmin(tt2,cc)
pcc <- 1- sum(status)/(n.cluster*(n.obs)) # percent censoring
print(pcc)
print(paste("simul corr(xmyb1,xmyb2) is ", round(corr(as.matrix(cbind(xmyb1,xmyb2))),2), " my.corr is ", my.
     corr, sep=""))
print(paste("simul var(xmyb1) is ", var(xmyb1), " var(xmyb2) is ", var(xmyb2),sep=""))
data2 <- data.frame(t ,status, x1,xmyb1, xmyb2,  eta.grp, cluster, z1, z2)  # about 20% censoring
my.data <- data2
#-------------------------------------------------------------------------------
# Name output pdf files  & data object names
#-------------------------------------------------------------------------------
output.pdf <-paste(kk,"final_Bivariate_","corr",my.corr*10,"_G",n.cluster,"_n",n.obs,"_r",ARMsteps,"xEM",
     EMsteps,"_Random_alpha",abs(T.alpha),".pdf", sep="")
outputxtable.txt <- paste(kk,"final_Bivariate_","corr",my.corr*10,"_G",n.cluster,"_n",n.obs,"_r",ARMsteps,"xEM"
     ,EMsteps,"_Random_alpha",abs(T.alpha),"Xtable.txt", sep="")
output.csv <-paste(kk,"final_Bivariate_","corr",my.corr*10,"_G",n.cluster,"_n",n.obs,"_r",ARMsteps,"xEM",
     EMsteps,"_Random_alpha",abs(T.alpha),".csv", sep="")
get_G.csv <-paste(kk,"final_Bivariate_","corr",my.corr*10,"_G",n.cluster,"_n",n.obs,"_r",ARMsteps,"xEM",EMsteps
     ,"_Random_alpha",abs(T.alpha),"get_G.csv", sep="")
```

```
output.name <- paste(kk,"final_Bivariate_","corr",my.corr*10,"_G",n.cluster,".n",n.obs,".r",ARMsteps,".xEM_
     Random_",EMsteps,"alpha",abs(T.alpha), sep="")
output_raw.csv <-paste(kk,"final_Bivariate_","corr",my.corr*10,"_G",n.cluster,"_n",n.obs,"_r",ARMsteps,"xEM",
     EMsteps,"_Random_alpha",abs(T.alpha),"_RAW.csv", sep="")
output.b.csv <-paste(kk,"final_Bivariate_","corr",my.corr*10,"_G",n.cluster,"_n",n.obs,"_r",ARMsteps,"xEM",
     EMsteps,"_Random_alpha",abs(T.alpha),"_b_vector.csv", sep="")
output.b1.csv <-paste(kk,"final_Bivariate_","corr",my.corr*10,"_G",n.cluster,"_n",n.obs,"_r",ARMsteps,"xEM",
     EMsteps,"_Random_alpha",abs(T.alpha),"_b1_vector.csv", sep="")
output.b2.csv <-paste(kk,"final_Bivariate_","corr",my.corr*10,"_G",n.cluster,"_n",n.obs,"_r",ARMsteps,"xEM",
     EMsteps,"_Random_alpha",abs(T.alpha),"_b2_vector.csv", sep="")
b.bT.by.site.csv <- paste(kk,"final_Bivariate_","corr",my.corr*10,"_G",n.cluster,"_n",n.obs,"_r",ARMsteps,"xEM"
     ,EMsteps,"_Random_alpha",abs(T.alpha),"_bbTbySite.csv", sep="")
output.hessian.csv <-paste(kk,"final_bivariate_","corr",my.corr*10,"_G",n.cluster,"_n",n.obs,"_r",ARMsteps,"xEM
     ",EMsteps,"_Random_alpha",abs(T.alpha),"_hessian.csv", sep="")
#-------------------------------------------------------------------------------
# Keep the raw dataset
#-------------------------------------------------------------------------------
write.csv(my.data , file=paste(directory,output_raw.csv,sep=""))
#-------------------------------------------------------------------------------
# Getting Z and W data matrices for fixed and random effects parameters
#-------------------------------------------------------------------------------
Z <- cbind(1,as.matrix(subset(my.data,select= c(x1, eta.grp))))
mymat2 <- rep(list(matrix(c(rep(1,n.obs/2),rep(0,n.obs/2),rep(0,n.obs/2),rep(1,n.obs/2)),n.obs,2)),n.cluster)
myW <-  bdiag(mymat2) # you need function 'bdiag'
newW <- matrix(rep(1:n.cluster,each=n.obs),nrow=nrow(my.data),ncol=1)
W <- myW  # (n.clusterxn.obs)x(n.cluster)  e.g. 200x10
#-----------------------------------------------------
#
# 1) Set initial values outside iterative function
#-------------------------------------------------------------------------------
init.out <-survreg(Surv(t,status)~ x1+eta.grp, data=data2, dist="loglogistic")
beta1<- init.out$coefficients[1] # intercept
beta2<- init.out$coefficients[2] # x1
beta3<- init.out$coefficients[3] # eta
alpha <- 0
sigma.mat <- matrix(c(1,0,0,1),2,2) # diag(2) # 2x2 identity matrix in R
random.b1 <- 0     # initial value for random effect b1
random.b2 <- 0   # initial value for random effect b2
init.beta1 <-0
init.beta2 <-0
init.beta3 <-0
init.alpha <-0
UPPER <- EMsteps  # number of loops
b.rho <-0
b.rho.x <- 0
  get.beta1 <- matrix(NA,nrow=UPPER,ncol=1)
  get.beta2 <- matrix(NA,nrow=UPPER,ncol=1)
  get.beta3 <- matrix(NA,nrow=UPPER,ncol=1)
  get.alpha <- matrix(NA,nrow=UPPER,ncol=1)
  get.sigma11 <- matrix(NA,nrow=UPPER,ncol=1)
  get.sigma22 <- matrix(NA, nrow=UPPER, ncol=1)
  get.sigma12 <- matrix(NA, nrow=UPPER, ncol=1)
  get.rho     <- matrix(NA, nrow=UPPER, ncol=1)
  get.rho.x   <- matrix(NA, nrow=UPPER, ncol=1)
  val.Q2 <- matrix(NA,nrow=UPPER,ncol=1)
  val.Q1 <- matrix(NA,nrow=UPPER,ncol=1)
  val.el <- matrix(NA,nrow=UPPER,ncol=1)
  val.E.gij <- vector("list", length=UPPER)
  get.sigma.mat <- vector("list",length=UPPER)
  get.G.mat <- matrix(NA, nrow=UPPER, ncol=4)
  # create a list to save up all random effects genereated
  get.b.vector <- vector("list",length=UPPER)
  get.b1.vector <- matrix(NA, nrow=UPPER, ncol=n.cluster) #
  get.b2.vector <- matrix(NA, nrow=UPPER, ncol=n.cluster) #
  sub.get.b1.vector <- matrix(NA, nrow=n.Gibbs, ncol=n.cluster) # added for p2 4/3/12
  sub.get.b2.vector <- matrix(NA, nrow=n.Gibbs, ncol=n.cluster) # added for p2 4/3/12
  x.all.b1.b2 <- matrix(NA, nrow=EMsteps, ncol=n.cluster)
  all.my.corr <- matrix(NA, nrow=EMsteps, ncol=1)
  mytest <- vector("list")
  b.bT.by.site <- vector("list")
  my.hessian <- vector("list")
#-------------------------------------------------------------------------------
#
# 2) For loop for Gibbs EM step
#-------------------------------------------------------------------------------
  for (k in (1:EMsteps)) {  # open big loop
  print(paste("Nested model rho=",my.corr," NUM.DATA=",kk,"--EM iteration ", k, "--n.cluster = ",n.cluster," n.
       obs = ",n.obs,sep=""))
  print(paste(" printing beta1 ", round(beta1,3)))
  print(paste(" printing beta2 ", round(beta2,3)))
  print(paste(" printing beta3 ", round(beta3,3)))
  print(paste(" printing alpha ", round(alpha,3)))
  print(paste(" printing sigma11 ", round(sigma.mat[1,1],3)))
  print(paste(" printing sigma22 ", round(sigma.mat[2,2],3)))
  print(paste(" printing sigma12 ", round(sigma.mat[1,2],3)))
#-------------------------------------------------------------------------------
# store all parameter values
#-------------------------------------------------------------------------------
  get.beta1[k,]<- beta1
```

```
  get.beta2[k,]<- beta2
  get.beta3[k,]<- beta3
  get.alpha[k,]<- alpha
  get.sigma11[k,] <- sigma.mat[1,1]
  get.sigma22[k,] <- sigma.mat[2,2]
  get.sigma12[k,] <- sigma.mat[1,2]
  init.theta <- c(get.beta1[k,],get.beta2[k,],get.beta3[k,],get.alpha[k,])
  get.rho[k,] <- b.rho
  get.rho.x[k,] <- b.rho.x
  print(paste(" printing get.rho ", round(get.rho[k,],2)," get.rho.x ", round(get.rho.x[k,],2)))
  get.sigma.mat[[k]] <- sigma.mat
#-------------------------------------------------------------------------
#  E-Step:
#-------------------------------------------------------------------------
  b.dist.z1 <- make.dist(1,'bdist1','plain("bdist1")(random.b1)',log.density=posterior.b.i.z.all.b1, grad.log.
       density=1, mean=0)
  b.dist.z2 <- make.dist(1,'bdist2','plain("bdist2")(random.b2)',log.density=posterior.b.i.z.all.b2, grad.log.
       density=1, mean=0)
#  beta1.dist <- make.dist(1,'betadist1','plain("betadist1")(beta1)',log.density=posterior.beta1, grad.log.
     density=1, mean=beta1, cov=NULL)
#  beta2.dist <- make.dist(1,'betadist2','plain("betadist2")(beta2)',log.density=posterior.beta2, grad.log.
     density=1, mean=beta2, cov=NULL)
#  beta3.dist <- make.dist(1,'betadist3','plain("betadist")(beta3)',log.density=posterior.beta3, grad.log.
     density=1, mean=beta3, cov=NULL)
#  alpha.dist <- make.dist(1,'alphadist','plain("alphadist")(alpha1)',log.density=posterior.alpha, grad.log.
     density=1, mean=alpha, cov=NULL)
#  print(paste("done making distributions"))
#-------------------------------------------------------------------------------------------------------
# Draw b (random effects) from full conditional distributions using Gibbs sampling b1|b2, and b2|b1
#-------------------------------------------------------------------------------------------------------
  sub.get.b1.vector <- matrix(NA, nrow=n.Gibbs, ncol=n.cluster) #
  sub.get.b2.vector <- matrix(NA, nrow=n.Gibbs, ncol=n.cluster) #
   for (gg in (1:n.Gibbs)) {
  b.samp <- vector("list",n.cluster)
  # draw b1 given b2
  for ( i in (1:n.cluster)) {
    ifelse(k >1, random.b2 <- get.b2.vector[k-1,i], random.b2)
    b.samp[[i]][[1]] <- arms.sample(b.dist.z1, x0=runif(1), sample.size=ARMsteps, tuning=5)$X
  }
  b1.vector.test <- sapply(b.samp, function(x) mean(x[[1]]))
      sub.get.b1.vector[gg,] <- b1.vector.test
  # draw b2 given b1
  for (i in (1:n.cluster)) {
     ifelse(k>1, random.b1 <- get.b1.vector[k-1,i], random.b1)  # update it for each site then goes into the
          function b.dist.z2
    b.samp[[i]][[2]] <- arms.sample(b.dist.z2, x0=runif(1), sample.size=ARMsteps, tuning=5)$X
  }
  b2.vector.test <- sapply(b.samp, function(x) mean(x[[2]]))
    sub.get.b2.vector[gg,] <- b2.vector.test
  } # END OF n.Gibbs steps
    get.b1.vector[k,] <- apply(sub.get.b1.vector,2, mean)
    get.b2.vector[k,] <- apply(sub.get.b2.vector,2, mean)
#-------------------------------------------------------------------------------------------------------
  b.vector.test <- vector("list", n.cluster)
  for ( i in (1:n.cluster)) {

    b.vector.test[[i]] <- sapply(b.samp[[i]], function(x) mean(x))

  } # each list is site. within each list each column is random effect 1 and 2 etc..

  b.vector <- do.call("rbind",b.vector.test) # thus each row is site with its random effects
  b <- as.vector(t(b.vector)) # this is 4x1 -- stack the rows of b.vector above
  # make a list to save 4x2 (n.cluster by #random effects) matrix for each iteration
  # use get.b.vector to calculate G matrix later
  get.b.vector[[k]] <- b.vector
#-------------------------------------------------------------------------
# M-step
#-------------------------------------------------------------------------
  my.output <- optim(init.theta, new.Q1, bb=b, method="L-BFGS-B" )
  beta1 <- my.output$par[1]
  beta2 <- my.output$par[2]
  beta3 <- my.output$par[3]
  alpha <- my.output$par[4]
  my.hessian[[k]] <- NA # my.output$hessian
#-------------------------------------------------------------------------
#  calculate E[G_{ij}|Y, \theta^(k)] function with given expectations from E-step
#-------------------------------------------------------------------------
val.E.gij[[k]] <- E.gij(beta1, beta2,beta3,alpha, old.beta=c(get.beta1[k,],get.beta2[k,], get.beta3[k,]), old.
     alpha=get.alpha[k,], bb=b)
#-------------------------------------------------------------------------
# update sigma matrix for z1 and z2 for unconstrained G var-cov matrix 2
#-------------------------------------------------------------------------
all.b1.mean <- apply((na.omit(get.b1.vector)), 2, mean)
all.b2.mean <- apply((na.omit(get.b2.vector)), 2, mean)
b1.row.mean <- matrix(apply(na.omit(get.b1.vector),1, mean), k, 1) # k = EMsteps
x.all.b1  <- matrix(apply(na.omit(get.b1.vector), 2, function(x)  x - b1.row.mean), k, n.cluster)
xx.all.b1 <- matrix(apply(na.omit(get.b1.vector), 2, function(x)  x ), k, n.cluster)
x.all.b1.mean <- apply(na.omit(x.all.b1), 2, mean)
all.b1.square   <- apply((na.omit(x.all.b1))^2, 2, mean) #
```

```
fresh.var.b1 <- sum(all.b1.square)/n.cluster
b2.row.mean <- matrix(apply(na.omit(get.b2.vector),1, mean), k, 1) # k = EMsteps
x.all.b2 <- matrix(apply(na.omit(get.b2.vector),  2, function(x)  x - b2.row.mean), k, n.cluster)
xx.all.b2 <- matrix(apply(na.omit(get.b2.vector), 2, function(x)  x ), k, n.cluster)
x.all.b2.mean <- apply(na.omit(x.all.b2), 2, mean)
all.b2.square   <- apply((na.omit(x.all.b2))^2, 2, mean) #
#all.b2.square   <- apply((na.omit(get.b2.vector))^2, 2, mean) #
fresh.var.b2 <- sum(all.b2.square)/n.cluster
b.rho <- corr(cbind(x.all.b1.mean,x.all.b2.mean))
if (b.rho.x>1) {n.cov.var <- b.rho*sqrt(fresh.var.b1)*sqrt(fresh.var.b2) } else {n.cov.var <- b.rho.x*sqrt(
      fresh.var.b1)*sqrt(fresh.var.b2)}
if(b.rho.x>1)  { b.rho.x<- 1 }
if(b.rho.x< (-1)) {b.rho.x<- (-1)}
n.cov.var <- mean(x.all.b1.mean*x.all.b2.mean)
b.rho.x <- n.cov.var/(sqrt(fresh.var.b1)*sqrt(fresh.var.b2))
my.G <- matrix( c(fresh.var.b1,n.cov.var,n.cov.var,fresh.var.b2),2,2,byrow=TRUE)
sigma.mat <- my.G
sigma11 <- my.G[1,1]
sigma12 <- my.G[1,2]
sigma22 <- my.G[2,2]
#-------------------------------------------------------------------------
#  calculate Q1
#-------------------------------------------------------------------------
 val.Q1[k,] <- calc.Q1(theta=c(beta1,beta2,beta3,alpha), bb=b)
# print(paste("printing value of Q1... ", val.Q1[k,],"for",k," iteration"))
#-------------------------------------------------------------------------
#  calculate Q2
#-------------------------------------------------------------------------
 val.Q2[k,] <- Q2(bb=b.vector, sigma)
# print(paste("printing value of Q2... ", val.Q2[k,],"for",k," iteration"))
#  print(paste("-----returning to loop..-------"))
#-------------------------------------------------------------------------
#  get el
#-------------------------------------------------------------------------
 val.el[k,] <- calc.el(theta=c(beta1,beta2,beta3,alpha), bb=b)
# print(paste("printing value of log1... ", val.el[k,],"for",k," iteration"))
} # End of EM loop
#-------------------------------------------------------------------------
#  Plot EM steps
#-------------------------------------------------------------------------
myx <- seq(1,UPPER,1)
ylab1.name <- expression(paste(hat(beta),"1"))
ylab2.name <- expression(paste(hat(beta),"2"))
ylab3.name <- expression(paste(hat(beta),"3"))
ylab4.name <- expression(paste(hat(alpha)))
ylab5.name <- expression(paste(hat(sigma),"11"))
ylab5b.name <- expression(paste(hat(sigma),"22"))
ylab5c.name <- expression(paste(hat(sigma),"12"))
ylab5d.name <- expression(paste(hat(rho)))
ylab6.name <- expression(paste(hat(plain(Q1))))
ylab7.name <- expression(paste(hat(plain(Q2))))

par(mfrow=c(5,2))
plot(myx , c(get.beta1[1:UPPER,]), type="l", xlab="EM steps", ylab=ylab1.name);
mtext(paste(kk," sim"," bivar ", "G=",n.cluster,"  n=",n.obs,"  alpha=", T.alpha, "  ARMS=",ARMsteps, "  EM
     steps=", UPPER," corr=",my.corr," Gibbs ",n.Gibbs,sep=""), side=3, line=1, outer=F, cex=0.7)
plot(myx , c(get.beta2[1:UPPER,]),type="l", xlab="EM steps", ylab=ylab2.name);
plot(myx , c(get.beta3[1:UPPER,]),type="l", xlab="EM steps", ylab=ylab3.name);
plot(myx , c(get.alpha[1:UPPER,]),type="l", xlab="EM steps", ylab=ylab4.name);
plot(myx , c(get.sigma11[1:UPPER,]),type="l", xlab="EM steps", ylab=ylab5.name);
plot(myx , c(get.sigma22[1:UPPER,]),type="l", xlab="EM steps", ylab=ylab5b.name);
plot(myx , c(get.rho.x[1:UPPER,]),type="l", xlab="EM steps", ylab=ylab5d.name);
plot(myx[-1] , c(val.Q1[2:UPPER,]),type="l", xlab="EM steps", ylab=ylab6.name);
plot(myx[-1] , c(val.Q2[2:UPPER,]),type="l", xlab="EM steps", ylab=ylab7.name);
# use option paper="a4r" for landscape
pdf(file=paste(directory,output.pdf,sep=""),onefile=TRUE ,paper="a4", height=11,width=8 )
par(mfrow=c(5,2))
plot(myx , c(get.beta1[1:UPPER,]), type="l", xlab="EM steps", ylab=ylab1.name);
mtext(paste(kk," sim"," bivar ", "G=",n.cluster,"  n=",n.obs,"  alpha=", T.alpha, "  ARMS=",ARMsteps, "  EM
     steps=", UPPER," corr=",my.corr," Gibbs ",n.Gibbs,sep=""), side=3, line=1, outer=F, cex=0.7)
plot(myx , c(get.beta2[1:UPPER,]),type="l", xlab="EM steps", ylab=ylab2.name);
plot(myx , c(get.beta3[1:UPPER,]),type="l", xlab="EM steps", ylab=ylab3.name);
plot(myx , c(get.alpha[1:UPPER,]),type="l", xlab="EM steps", ylab=ylab4.name);
plot(myx , c(get.sigma11[1:UPPER,]),type="l", xlab="EM steps", ylab=ylab5.name);
plot(myx , c(get.sigma22[1:UPPER,]),type="l", xlab="EM steps", ylab=ylab5b.name);
plot(myx , c(get.rho.x[1:UPPER,]),type="l", xlab="EM steps", ylab=ylab5d.name);
plot(myx[-1] , c(val.Q1[2:UPPER,]),type="l", xlab="EM steps", ylab=ylab6.name);
plot(myx[-1] , c(val.Q2[2:UPPER,]),type="l", xlab="EM steps", ylab=ylab7.name);
dev.off()
#-------------------------------------------------------------------------
# save results in .csv file
#-------------------------------------------------------------------------
all.matrix.G40.n50.a1.0em1000 <- cbind(get.beta1, get.beta2, get.beta3,get.alpha, get.sigma11,get.sigma22, get.
     sigma12,get.rho, get.rho.x, val.Q1, val.Q2)
colnames(all.matrix.G40.n50.a1.0em1000) <- c("beta1","beta2","beta3","alpha","sigma11","sigma22","sigma12", "
     rho","rho.x", "Q1","Q2")
write.csv(all.matrix.G40.n50.a1.0em1000, file=paste(directory,output.csv,sep=""))
write.csv(get.G.mat, file=paste(directory, get_G.csv, sep=""))
# save get.b.vector list
```

```
dput(get.b.vector,file=paste(directory,output.b.csv,sep=""))
# save each b1.vector and b2.vector to separate files?
write.csv(get.b1.vector, file=paste(directory, output.b1.csv, sep=""))
write.csv(get.b2.vector, file=paste(directory, output.b2.csv, sep=""))
# save b.bT.by.site list which contains results of b1%*% t(b1) for each site
dput(b.bT.by.site,file=paste(directory,b.bT.by.site.csv,sep=""))
myresults <- all.matrix.G40.n50.a1.0em1000
#save my.hessian
dput(my.hessian, file=paste(directory, output.hessian.csv, sep=""))
#-------------------------------------------------------------------------
#  gather the results for LaTex table
#-------------------------------------------------------------------------
# true.value in the order of beta1, beta2, eta, alpha, sigma11, sigma22, corr
 my.results.G40.n50.a1.0em1000 <- cbind(get.beta1, get.beta2, get.beta3, get.alpha, get.sigma11, get.sigma22,
      get.sigma12,get.rho, get.rho.x)
 my.results.Q <- cbind(val.Q1,val.Q2)
 mean.q  <-apply(my.results.Q[-c(1:initial),], 2, function(x) mean(x))
 mean.el <- mean(val.el[-c(1:initial),])
 mean.get.G <- apply(get.G.mat,2,mean)
 mean.param.estimates <- apply(my.results.G40.n50.a1.0em1000[-c(1:initial),], 2, function(x) mean(x))
 empirical.bias       <- mean.param.estimates - true.value
 empirical.SD         <- apply(my.results.G40.n50.a1.0em1000[-c(1:initial),], 2, function(x) sd(x))
 empirical.SE         <- apply(my.results.G40.n50.a1.0em1000[-c(1:initial),], 2, function(x) sd(x))
 parameters         <- c("beta0", "beta1" ,"eta","alpha","sigma11","sigma22","sigma12","rho","rho.x")
 test.table.G40.n50.a1.0em1000 <- cbind(true.value, mean.param.estimates, empirical.bias, empirical.SD)
 final.table.G40.n50.a1.0em1000 <- data.frame(parameters, test.table.G40.n50.a1.0em1000)
#final.table.G40.n50.a1.0em1000

sink(file=paste(directory,outputxtable.txt,sep=""))
  print(paste("Bivariate_",n.cluster,"_clusters_",n.obs,"_obs",sep=""))
  options(scipen=4)
  print(pcc)
  print(paste("simul corr(xmyb1,xmyb2) is ", round(corr(as.matrix(cbind(xmyb1,xmyb2))),2), " my.corr is ", my.
       corr, sep=""))
  print(paste("simul var(xmyb1) is ", var(xmyb1), " var(xmyb2) is ", var(xmyb2),sep=""))
  print(paste("mean.get.G ", mean.get.G))
  print(final.table.G40.n50.a1.0em1000)
  print(paste("Bivariate model with G=", n.cluster," n=", n.obs,"n.Gibbs=", n.Gibbs, " EM=", EMsteps,sep=""))
  xtable( final.table.G40.n50.a1.0em1000, digits=4)
sink()
#-------------------------------------------------------------------------
#  gather the results for each NUM.DATA run
#-------------------------------------------------------------------------
  simul.summary[kk,1] <- mean.param.estimates[1]  #beta1
  simul.summary[kk,2] <- mean.param.estimates[2]  #beta2
  simul.summary[kk,3] <- mean.param.estimates[3]  #beta3
  simul.summary[kk,4] <- mean.param.estimates[4]  #alpha
  simul.summary[kk,5] <- mean.param.estimates[5]  #sigma11
  simul.summary[kk,6] <- mean.param.estimates[6]  #sigma22
  simul.summary[kk,7] <- mean.param.estimates[7]  #sigma12
  simul.summary[kk,8] <- mean.param.estimates[8]  #get.rho
  simul.summary[kk,9] <- mean.param.estimates[9]  #get.rho.x
  simul.summary[kk,10] <-  empirical.SD[1]  #beta1
  simul.summary[kk,11] <-  empirical.SD[2]  #beta2
  simul.summary[kk,12] <-  empirical.SD[3]  #beta3
  simul.summary[kk,13] <-  empirical.SD[4]  #alpha
  simul.summary[kk,14] <-  empirical.SD[5]  #sigma11
  simul.summary[kk,15] <-  empirical.SD[6]  #sigma22
  simul.summary[kk,16] <-  empirical.SD[7]  #sigma12
  simul.summary[kk,17] <-  empirical.SD[8]  #get.rho
  simul.summary[kk,18] <-  empirical.SD[9]  #get.rho.x
  simul.summary[kk,19] <- mean.q[1]  #q1
  simul.summary[kk,20] <- mean.q[2]  #q2
  simul.summary[kk,21] <- round(pcc,2)  #pcc
  simul.summary[kk,22] <- my.seed
  simul.summary[kk,23] <- round(var(xmyb1),2)
  simul.summary[kk,24] <- round(var(xmyb2),2)
  simul.summary[kk,25] <-  round(corr(as.matrix(cbind(xmyb1,xmyb2))),2)
  simul.summary[kk,26] <- mean.el
  }  ##END OF BIG NUM.DATA RUN (kk)
colnames(simul.summary) <- c("beta0","beta1","eta","alpha","sigma11","sigma22","sigma12","get.rho","get.rho.x",
     "beta0.sd","beta1.sd","eta.sd","alpha.sd","sigma11.sd","sigma22.sd","sigma12.sd","get.rho.se","get.rho.x.se
     ","Q1","Q2","PCC","my.seed", "var.myb1","var.myb2","corr.myb1b2","mean.log1")
#write.csv( simul.summary , file=paste(directory,"Simulation_table_out_", NUM.DATA, "RUNS.csv",sep=""))
write.csv( simul.summary , file=paste(directory,"G",n.cluster,"_n.obs",n.obs,"_Simulation_table_out_", NUM.DATA
     , "RUNS.csv",sep=""))
#-------------------------------------------------
#
#-------------------------------------------------
end.time <- date()
sink(file=paste(directory,"Final_summary_bivariate",n.cluster,"cluster_",n.obs,"obs_", EMsteps,"EMsteps.txt",
     sep=""))
  print(paste("Bivariate_",n.cluster,"_clusters_",n.obs,"_obs",sep=""))
  options(scipen=4)
  avg.summary <- as.matrix(apply(simul.summary, 2, function(x) mean(x)))
  avg.summary
  x1.c <- strptime(begin.time, "%a %b %d %H:%M:%S %Y")
  x2.c <- strptime(end.time, "%a %b %d %H:%M:%S %Y")
  difftime(x2.c, x1.c, units='mins')
```

```
  difftime(x2.c, x1.c, units='hours')
  difftime(x2.c, x1.c, units='days')
 print(paste("n.cluster=", n.cluster, " n.obs=", n.obs, " my.corr=", my.corr, " T.alpha=", T.alpha, " theta=",
      my.theta1," EM=", EMsteps, " ARMS=", ARMsteps, " n.Gibbs=", n.Gibbs, sep=""))
 hrs <- difftime(x2.c, x1.c, units='hours')
 req.hours <- (hrs/NUM.DATA)*200
 print(paste("estimated time required to run NUM.DATA=200 is ",  round(req.hours,2), " hours"))
 print(paste("NUM.DATA=",NUM.DATA))
  #print(paste("simul corr(xmyb1,xmyb2) is ", round(corr(as.matrix(cbind(xmyb1,xmyb2))),2), " my.corr is ", my.
       corr, sep=""))
  #print(paste("simul var(xmyb1) is ", var(xmyb1), " var(xmyb2) is ", var(xmyb2),sep=""))
  #print(paste("mean.get.G ", mean.get.G))
sink()

options(scipen=4)
avg.summary <- as.matrix(apply(simul.summary, 2, function(x) mean(x)))
avg.summary
x1.c <- strptime(begin.time, "%a %b %d %H:%M:%S %Y")
x2.c <- strptime(end.time, "%a %b %d %H:%M:%S %Y")
difftime(x2.c, x1.c, units='secs')
difftime(x2.c, x1.c, units='mins')
difftime(x2.c, x1.c, units='hours')
print(paste("n.cluster=", n.cluster, " n.obs=", n.obs, " my.corr=", my.corr, " T.alpha=", T.alpha, " theta=",
     my.theta1," EM=", EMsteps, " ARMS=", ARMsteps, " n.Gibbs=", n.Gibbs, sep=""))
hrs <- difftime(x2.c, x1.c, units='hours')
req.hours <- (hrs/NUM.DATA)*200
print(paste("estimated time required to run NUM.DATA=200 is ",  round(req.hours,2), " hours"))
```

./nested_AFT_model_with_random_effects_simulation.R

```
################################################################################
# Variance estimation by Louis method after EM
################################################################################
#remove(list=ls(all=TRUE))
library(lattice)
library(mcmc)
library(coda)
library(lattice)
library(MASS)
library(MCMCpack)
library(mvtnorm)
library(SamplerCompare) # gives arms.sample function
library(splines)
library(survival)
library(smoothSurv)
library(bayesSurv)
library(xtable)
library(boot)
#-------------------------------------------------------------------------
# Run a function that creates a diagonal matrix
#-------------------------------------------------------------------------
bdiag <- function(x){
     if(!is.list(x)) stop("x not a list")
     n <- length(x)
     if(n==0) return(NULL)
     x <- lapply(x, function(y) if(length(y)) as.matrix(y) else
stop("Zero-length component in x"))
     d <- array(unlist(lapply(x, dim)), c(2, n))
     rr <- d[1,]
     cc <- d[2,]
     rsum <- sum(rr)
     csum <- sum(cc)
     out <- array(0, c(rsum, csum))
     ind <- array(0, c(4, n))
     rcum <- cumsum(rr)
     ccum <- cumsum(cc)
     ind[1,-1] <- rcum[-n]
     ind[2,] <- rcum
     ind[3,-1] <- ccum[-n]
     ind[4,] <- ccum
     imat <- array(1:(rsum * csum), c(rsum, csum))
     iuse <- apply(ind, 2, function(y, imat) imat[(y[1]+1):y[2],
(y[3]+1):y[4]], imat=imat)
     iuse <- as.vector(unlist(iuse))
     out[iuse] <- unlist(x)
     return(out)
 }
#-------------------------------------------------------------------------
# Change the input directory where data are
#-------------------------------------------------------------------------
directory  <-"c:\\_dissertation\\outputs\\BivariateModel\\optim\\nested\\corr06\\G50n30\\"  # NEED TO CHANGE
begin.time <- date()
#-------------------------------------------------------------------------
# Simulation parameters --same as earliler
#-------------------------------------------------------------------------
  NUM.DATA <- 200
```

```
  num.par <- 7 # number of parameters
  n.cluster  <- 60   # CHANGE HERE  overall number of clusters
  n.obs      <- 30   # CHANGE HERE  n.obs/2 per each sub-cluster
  cluster    <- c(rep(1:n.cluster,each=n.obs))
  n.sub.cluster <- 2
  EMsteps   <- 250
  initial   <- 50    # first 50 estimates will not be included to estimate parameter means
  ARMsteps  <- 1
  n.Gibbs   <- 1
  T.alpha   <- 1
  my.theta1 <- 1  # so that sigma11 should be 1
  my.corr   <- 0.6
  my.sigma  <- matrix(c(1,my.corr,my.corr,1),2,2)
  T.eta.grp <- 1
  true.value          <- c(1,1,T.eta.grp,T.alpha,my.sigma[1,1], my.sigma[2,2], my.sigma[1,2])
  louis.summary <- matrix(NA,nrow=NUM.DATA,ncol=num.par)
for (kk in (1:NUM.DATA)) {
#-------------------------------------------------------------------------
# Name output pdf files  & data object names
#-------------------------------------------------------------------------
output.pdf <-paste(kk,"final_Bivariate_","corr",my.corr*10,"_G",n.cluster,"_n",n.obs,"_r",ARMsteps,"xEM",
     EMsteps,"_Random_alpha",abs(T.alpha),".pdf", sep="")
outputxtable.txt <- paste(kk,"final_Bivariate_","corr",my.corr*10,"_G",n.cluster,"_n",n.obs,"_r",ARMsteps,"xEM"
     ,EMsteps,"_Random_alpha",abs(T.alpha),"Xtable.txt", sep="")
output.csv <-paste(kk,"final_Bivariate_","corr",my.corr*10,"_G",n.cluster,"_n",n.obs,"_r",ARMsteps,"xEM",
     EMsteps,"_Random_alpha",abs(T.alpha),".csv", sep="")
get_G.csv <-paste(kk,"final_Bivariate_","corr",my.corr*10,"_G",n.cluster,"_n",n.obs,"_r",ARMsteps,"xEM",EMsteps
     ,"_Random_alpha",abs(T.alpha),"get_G.csv", sep="")
output.name <- paste(kk,"final_Bivariate_","corr",my.corr*10,"_G",n.cluster,".n",n.obs,".r",ARMsteps,".xEM_
     Random_",EMsteps,"alpha",abs(T.alpha), sep="")
output_raw.csv <-paste(kk,"final_Bivariate_","corr",my.corr*10,"_G",n.cluster,"_n",n.obs,"_r",ARMsteps,"xEM",
     EMsteps,"_Random_alpha",abs(T.alpha),"_RAW.csv", sep="")
output.b.csv <-paste(kk,"final_Bivariate_","corr",my.corr*10,"_G",n.cluster,"_n",n.obs,"_r",ARMsteps,"xEM",
     EMsteps,"_Random_alpha",abs(T.alpha),"_b_vector.csv", sep="")
output.b1.csv <-paste(kk,"final_Bivariate_","corr",my.corr*10,"_G",n.cluster,"_n",n.obs,"_r",ARMsteps,"xEM",
     EMsteps,"_Random_alpha",abs(T.alpha),"_b1_vector.csv", sep="")
output.b2.csv <-paste(kk,"final_Bivariate_","corr",my.corr*10,"_G",n.cluster,"_n",n.obs,"_r",ARMsteps,"xEM",
     EMsteps,"_Random_alpha",abs(T.alpha),"_b2_vector.csv", sep="")
b.bT.by.site.csv <- paste(kk,"final_Bivariate_","corr",my.corr*10,"_G",n.cluster,"_n",n.obs,"_r",ARMsteps,"xEM"
     ,EMsteps,"_Random_alpha",abs(T.alpha),"_bbTbySite.csv", sep="")
output.hessian.csv <-paste(kk,"final_bivariate_","corr",my.corr*10,"_G",n.cluster,"_n",n.obs,"_r",ARMsteps,"xEM
     ",EMsteps,"_Random_alpha",abs(T.alpha),"_hessian.csv", sep="")
#-------------------------------------------------------------------------------
# Get the simulated data
#-------------------------------------------------------------------------------
data2 <- read.csv(file=paste(directory,output_raw.csv,sep=""), header=TRUE, sep=",")
new.data <- data2
my.data  <- data2
#-------------------------------------------------------------------------
# Getting Z and W data matrices for fixed and random effects parameters
#-------------------------------------------------------------------------
Z <- cbind(1,as.matrix(subset(my.data,select= c(x1, eta.grp))))
mymat2 <- rep(list(matrix(c(rep(1,n.obs/2),rep(0,n.obs/2),rep(0,n.obs/2),rep(1,n.obs/2)),n.obs,2)),n.cluster)
myW <-  bdiag(mymat2) # you need function 'bdiag'
newW <- matrix(rep(1:n.cluster,each=n.obs),nrow=nrow(my.data),ncol=1)
W <- myW  # (n.clusterxn.obs)x(n.cluster)  e.g. 200x10
#-------------------------------------------------------------------------------
# GET beta estimates from the model and the output from each simulation
#-------------------------------------------------------------------------------
xmy.results <- read.csv(file=paste(directory, output.csv, sep=""), header=TRUE)
my.results <- xmy.results[,2:10] # only selecting beta1 beta2 eta alpha sigma11 sigma22 sigma12 rho rho.x
     parameters
mean.param.estimates <- apply(my.results[-c(1:initial),], 2, function(x) mean(x))
h.beta1 <- mean.param.estimates[1]
h.beta2 <- mean.param.estimates[2]
h.beta3 <- mean.param.estimates[3]
h.alpha <- mean.param.estimates[4]
h.sigma11 <- sqrt(mean.param.estimates[5])
h.sigma22 <- sqrt(mean.param.estimates[6])
h.rho.x     <- (mean.param.estimates[8]) # for rho
all.mean.param.estimates <- c(h.beta1,h.beta2,h.beta3, h.alpha,h.sigma11^2,h.sigma22^2,h.rho.x)
h.beta.all <- matrix(c(h.beta1, h.beta2, h.beta3),3,1)
beta <- h.beta.all
#-------------------------------------------------------------------------------
# Need b1 and b2 vectors.  These vectors changes.
#-------------------------------------------------------------------------------
b1.vector.out <- read.csv(file=paste(directory,output.b1.csv, sep=""),header=TRUE)
b2.vector.out <- read.csv(file=paste(directory,output.b2.csv, sep=""),header=TRUE)
b1.out <- b1.vector.out[-c(1:initial),-1] # remove first 50 rows and first column
b2.out <- b2.vector.out[-c(1:initial),-1] # remove first 50 rows and first column
#^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
# Calculate second derivative and score matrix for each b matrix
# Save the results in the lists
#^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
score.score.T <- vector("list", length=(EMsteps-initial))
score         <- vector("list", length=(EMsteps-initial))
second.deriv  <- vector("list", length=(EMsteps-initial))
x.Louis.v     <- vector("list", length=(EMsteps-initial))
for (k in (1:(EMsteps-initial))) {
```

```
get.b1 <- b1.out[k,]
get.b2 <- b2.out[k,]
both.b <- as.matrix(rbind(get.b1,get.b2))
b <- as.vector(both.b) # check this b again with the rest of the codes go back to other programste
print(paste("--NUM.DATA=",kk,"---------iteration ", k))
new.data <- my.data
W.b   <- W %*% b       # 200x10 time 10x1 genereate 200x1 W.b matrix
Z.beta<- Z %*% beta    # 200x3 times 3x1 generates 200x1 Z.beta matrix
logT  <- log(new.data$t)
T     <- new.data$t
delta <- as.matrix(new.data$status, nrow(new.data),ncol=1)
e.A <- exp(h.alpha+logT - Z.beta - W.b)  # this will change per b values as well
#^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
#  Define derivatives for the score
#^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
Z.1 <- matrix(Z[,1],nrow(Z),1)
Z.2 <- matrix(Z[,2],nrow(Z),1)
Z.3 <- matrix(Z[,3],nrow(Z),1)
#### this is working version 1 -- small numbers
D.beta1 <- (-1)*sum( Z.1*(delta - exp(-h.alpha)*e.A)/(1+ e.A))
D.beta2 <- (-1)*sum( Z.2*(delta - exp(-h.alpha)*e.A)/(1+ e.A))
D.beta3 <- (-1)*sum( Z.3*(delta - exp(-h.alpha)*e.A)/(1+ e.A))
D.alpha <-  (-1)*sum( (-1)*exp(-h.alpha)*log(1+e.A) + ((delta+ exp(-h.alpha))*((e.A)/(1+e.A))))
#--------------------------------------------------------------------------------
# h.sigma11, h.sigma22 and h.rho.x parameters
#--------------------------------------------------------------------------------
D.sigma11 <- (-1/2)*(n.cluster/(h.sigma11^2)) + sum(get.b1^2)/(2*(1-h.rho.x^2)*h.sigma11^4)-(1/2)*(sum(get.b1*get.b2)*h.rho.x)/((1-h.rho.x^2)*h.sigma22*(h.sigma11^3))
D.sigma22 <- (-1/2)*(n.cluster/(h.sigma22^2)) + sum(get.b2^2)/(2*(1-h.rho.x^2)*h.sigma22^4)-(1/2)*(sum(get.b1*get.b2)*h.rho.x)/((1-h.rho.x^2)*h.sigma11*(h.sigma22^3))
D.rho.x  <-(n.cluster*n.obs*h.rho.x)/(1-h.rho.x^2) - (sum(get.b1^2)/(h.sigma11^2) + sum(get.b2^2)/(h.sigma22^2))*(h.rho.x*(1-h.rho.x^2)^(-2)) + (sum(get.b1*get.b2)/(h.sigma11*h.sigma22))*((1+h.rho.x^2)/(1-h.rho.x^2)^2)
#--------------------------------------------------------------------------------
# the individual score
#--------------------------------------------------------------------------------
score.theta  <- matrix(c(D.beta1,D.beta2,D.beta3,D.alpha,D.sigma11,D.sigma22,D.rho.x),7,1)
score.score.T[[k]] <- score.theta %*% t(score.theta) # store each matrix into a list
score[[k]]         <- score.theta  # this is 5x1 matrix
#^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
# Second derivative matrix
#^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
# this is working version now
DD.beta1 <- (-1)*sum((Z.1^2 * e.A * (exp(-h.alpha)+delta))/((1+e.A)^2))
DD.beta2 <- (-1)*sum((Z.2^2 * e.A * (exp(-h.alpha)+delta))/((1+e.A)^2))
DD.beta3 <- (-1)*sum((Z.3^2 * e.A * (exp(-h.alpha)+delta))/((1+e.A)^2))
DD.alpha <- (-1)*sum(exp(-h.alpha)*log(1+e.A) - (e.A*exp(-h.alpha)/(1+e.A))+ (e.A*(delta-exp(-h.alpha)*(e.A))/((1+e.A)^2)))
DD.sigma11 <- (1/2)*(n.cluster/(h.sigma11^4))-sum(get.b1^2)/((1-h.rho.x^2)*h.sigma11^6)+ (3/4)*(sum(get.b1*get.b2)*h.rho.x)/((1-h.rho.x^2)*h.sigma22*h.sigma11^5)
DD.sigma22 <- (1/2)*(n.cluster/(h.sigma22^4))-sum(get.b2^2)/((1-h.rho.x^2)*h.sigma22^6)+ (3/4)*(sum(get.b1*get.b2)*h.rho.x)/((1-h.rho.x^2)*h.sigma11*h.sigma22^5)
a.1 <- n.cluster*n.obs*(1+h.rho.x^2)/((1-h.rho.x^2)^2)
b.1 <- (-1)*(sum(get.b1^2)/(h.sigma11^2) + sum(get.b2^2)/(h.sigma22^2) )*(( (1-h.rho.x^2)^2 + 4*h.rho.x*(1-h.rho.x^2))/(1-h.rho.x^2)^4)
c.1 <- (sum(get.b1*get.b2)/(h.sigma11*h.sigma22))*( ( 2*h.rho.x*(1-h.rho.x^2)^2 + 4*h.rho.x*(1-h.rho.x^2)*(1+h.rho.x^2))/(1-h.rho.x^2)^4)
DD.rho.x <- a.1 + b.1 + c.1
#2nd trial
DD.beta12 <- 0  #(-1)*(-1)*sum((Z.12 %*% e.A %*% t(exp(-h.alpha)+delta) ))/sum((1+e.A)%*%t(1+e.A))
DD.beta13 <- 0  #(-1)*(-1)*sum((Z.13 %*% e.A %*% t(exp(-h.alpha)+delta) ))/sum((1+e.A)%*%t(1+e.A))
DD.beta23 <- 0  # (-1)*(-1)*sum((Z.23 %*% e.A %*% t(exp(-h.alpha)+delta) ))/sum((1+e.A)%*%t(1+e.A))
DD.beta1.alpha <- (-1)*sum((-1)*(Z.1*e.A*(delta - exp(-h.alpha)*e.A))/(1+e.A)^2 )
DD.beta2.alpha <- (-1)*sum((-1)*(Z.2*e.A*(delta - exp(-h.alpha)*e.A))/(1+e.A)^2 )
DD.beta3.alpha <- (-1)*sum((-1)*(Z.3*e.A*(delta - exp(-h.alpha)*e.A))/(1+e.A)^2 )
DD.beta1.sigma11 <- 0
DD.beta2.sigma11 <- 0
DD.beta3.sigma11 <- 0
DD.alpha.sigma11 <- 0
DD.beta1.sigma22 <- 0
DD.beta2.sigma22 <- 0
DD.beta3.sigma22 <- 0
DD.alpha.sigma22 <- 0
#DD.sigma11.22 <-0
partA.1 <- sum(get.b1^2)*(h.rho.x*(1-h.rho.x^2)/(h.sigma11^4))
partA.2 <- (-1/2)*(sum(get.b1*get.b2)/(h.sigma22*(h.sigma11^3)))*( (1+h.rho.x^2)/((1-h.rho.x^2)^2))
DD.sigma11.rho <-  partA.1 + partA.2
partB.1 <-sum(get.b2^2)*(h.rho.x*(1-h.rho.x^2)/(h.sigma22^4))
partB.2 <-(-1/2)*(sum(get.b1*get.b2)/(h.sigma11*(h.sigma22^3)))*( (1+h.rho.x^2)/((1-h.rho.x^2)^2))
DD.sigma22.rho <- partB.1+partB.2
DD.sigma11.22 <- (1/4)*(sum(get.b1*get.b2)*h.rho.x)/((1-h.rho.x^2)*(h.sigma11^3)*(h.sigma22^3))
DD.sigma22.11 <- DD.sigma11.22 # same
x.second.deriv <- rbind( c(DD.beta1     , DD.beta12     , DD.beta13   , DD.beta1.alpha  , DD.beta1.sigma11,DD.beta1.sigma22, 0),
            c(DD.beta12    , DD.beta2      , DD.beta23   , DD.beta2.alpha  , DD.beta2.sigma11,DD.beta2.sigma22, 0),
            c(DD.beta13      , DD.beta12     , DD.beta3    , DD.beta3.alpha  , DD.beta3.sigma11,DD.beta3.sigma22, 0),
            c(DD.beta1.alpha , DD.beta2.alpha    , DD.beta3.alpha  , DD.alpha        , DD.alpha.sigma11,DD.alpha.sigma22, 0),
```

```
211             c(DD.beta1.sigma11 , DD.beta2.sigma11, DD.beta3.sigma11, DD.alpha.sigma11, DD.sigma11      ,DD.
                    sigma11.22   ,  DD.sigma11.rho) ,
212             c(DD.beta1.sigma22 , DD.beta2.sigma22, DD.beta3.sigma22, DD.alpha.sigma22, DD.sigma11.22   ,DD.
                    sigma22      , DD.sigma22.rho)  ,
213             c(      0,                  0,         0,    0       , DD.sigma11.rho  ,DD.sigma22.rho  , DD.rho
                    .x  ))
214 #print(paste("calculating second derivative.."))
215 #print(x.second.deriv)
216 second.deriv[[k]] <- as.matrix(x.second.deriv)
217   } # End of [k] Loop calculation
218 #-------------------------------------------------------------------------------
219 # The average score over b
220 #-------------------------------------------------------------------------------
221  avg.score.score.T <- matrix(NA,num.par,num.par)
222   for (i in (1:num.par))
223     for (j in (1:num.par))
224     {
225   avg.score.score.T[i,j] <-  mean(sapply(score.score.T, function(x) x[i,j]))
226     }
227   x.avg.score  <- matrix(NA, num.par,1)
228   for (i in (1:num.par))
229      { x.avg.score[i,1] <- mean(sapply(score, function(x) x[i,1])) }
230   avg.score <- x.avg.score %*% t(x.avg.score)
231 #-------------------------------------------------------------------------------
232 # The expected second derivative matrix
233 #-------------------------------------------------------------------------------
234  avg.second.deriv <- matrix(NA,num.par,num.par)
235   for (i in (1:num.par))
236     for (j in (1:num.par))
237     {
238   avg.second.deriv[i,j] <-  mean(sapply(second.deriv, function(x) x[i,j]))
239     }
240 #   print(paste("printing expected second derivative matrix..."))
241 #   avg.second.deriv
242 #-------------------------------------------------------------------------------
243 # Calculate variance-covariance I(theta) matrix using Louis formula
244 #-------------------------------------------------------------------------------
245 options(scipen=10)
246 # print(paste("printing difference of the average.."))
247 Louis.v <- (-1)*avg.second.deriv - (avg.score.score.T - avg.score)
248 Louis.v
249 Louis.var.cov <- solve(Louis.v)
250 Louis.var.cov
251 num.param <- num.par
252 louis.var.pos <- vector("list",EMsteps-initial)
253 louis.inv.pos <- vector("list",EMsteps-initial)
254 louis.inv.diag <- matrix(NA, EMsteps-initial, num.param)
255 louis.se.pos   <- matrix(NA, EMsteps-initial, num.param)
256 for (ii in (1:(EMsteps-initial))) {
257 louis.var.pos[[ii]] <- (-1)*second.deriv[[ii]] - score.score.T[[ii]]
258 louis.inv.pos[[ii]]  <- solve(louis.var.pos[[ii]])
259 louis.inv.diag[ii,] <- diag(louis.inv.pos[[ii]])
260 louis.se.pos[ii,]   <-  t(as.matrix(sqrt(louis.inv.diag[ii,])))
261 }
262 # counting number of missings in louis.se.pos[ii,]
263 my.count <- apply(louis.se.pos, 1, function(x) sum(is.na(x))) #/ ncol(louis.se.pos) * 100
264 # drop the row with negative iterations
265 louis.se.pos2 <- louis.se.pos[apply(louis.se.pos, 1, function(x)!any(is.na(x))), , drop=F]
266 #-------------------------------------------------------------------------------
267 # print out final Louis SE
268 #-------------------------------------------------------------------------------
269 final.louis.se <- apply(louis.se.pos2, 2, function(x) mean(x))
270 #  print(paste("final Louis SE..",sep=""))
271 #  final.louis.se
272 #-------------------------------------------------------------------------------
273 # print out final Louis SE
274 #-------------------------------------------------------------------------------
275   final.louis.se <- apply(louis.se.pos2, 2, function(x) mean(x))
276   #print(paste("final Louis SE..",sep=""))
277   #print(final.louis.se)
278   louis.summary[kk,1] <- final.louis.se[1]
279   louis.summary[kk,2] <- final.louis.se[2]
280   louis.summary[kk,3] <- final.louis.se[3]
281   louis.summary[kk,4] <- final.louis.se[4]
282   louis.summary[kk,5] <- final.louis.se[5]
283   louis.summary[kk,6] <- final.louis.se[6]
284   louis.summary[kk,7] <- final.louis.se[7]
285 } # END OF KK iterations
286 colnames(louis.summary) <- c("beta0.se","beta1.se","eta.se","alpha.se","sigma11.se","sigma22.se","get.rho.se")
287 write.csv( louis.summary , file=paste(directory,"corr",my.corr*10,"_G",n.cluster,"_n.obs",n.obs,"_Simulation_
        table_out_", NUM.DATA, "RUNS_Louis_SE.csv",sep=""))
288 end.time <- date()
289  x1.c <- strptime(begin.time, "%a %b %d %H:%M:%S %Y")
290  x2.c <- strptime(end.time, "%a %b %d %H:%M:%S %Y")
291  difftime(x2.c, x1.c, units='secs')
292  difftime(x2.c, x1.c, units='mins')
293  difftime(x2.c, x1.c, units='hours')
294 #-------------------------------------------------------------------------------
295 #-------------------------------------------------------------------------------
```

```
sim.out <- read.csv( file=paste(directory,"G",n.cluster,"_n.obs",n.obs,"_Simulation_table_out_", NUM.DATA, "
     RUNS.csv",sep=""))
xresults.out <- sim.out[,c(2:7,9)]
xlouis.out <- louis.summary
all.out <- na.omit(cbind(xresults.out, xlouis.out))
results.out <- all.out[,c(1:7)]
louis.out   <- all.out[,c(8:14)]
mean.out <- apply(results.out, 2,mean)
median.out <-apply(results.out, 2,median)
sd.out   <- apply(results.out, 2, sd)
avg.se.out <- apply(louis.out, 2, mean)
bias.out   <- mean.out - true.value
# percent.bias updated
percent.bias <-  ifelse(true.value==0, (mean.out-true.value)*100,((mean.out-true.value)/true.value)*100)
mse.out <-   apply( apply(results.out, 2, function(y)   (y-1)^2), 2, sum)/(NUM.DATA)
t05 <- qt(0.975, n.cluster*n.obs-1)
c.louis <- as.data.frame(louis.out)
c.beta0 <- 100*(sum((results.out$beta0-t05*c.louis$beta0.se <= true.value[1]) & (results.out$beta0+t05*c.louis$
     beta0.se >= true.value[1])))/NUM.DATA
c.beta1 <- 100*(sum((results.out$beta1-t05*c.louis$beta1.se <= true.value[2]) & (results.out$beta1+t05*c.louis$
     beta1.se >= true.value[2])))/NUM.DATA
c.eta   <- 100*(sum((results.out$eta-t05*c.louis$eta.se <= true.value[3]) & (results.out$eta+t05*c.louis$eta.se
      >= true.value[3])))/NUM.DATA
c.alpha <- 100*(sum((results.out$alpha-t05*c.louis$alpha.se <= true.value[4]) & (results.out$alpha+t05*c.louis$
     alpha.se >= true.value[4])))/NUM.DATA
c.theta1 <- 100*(sum((results.out$sigma11-t05*c.louis$sigma11.se <= true.value[5]) & (results.out$sigma11+t05*c
     .louis$sigma11.se >= true.value[5])))/NUM.DATA
c.theta2 <- 100*(sum((results.out$sigma22-t05*c.louis$sigma22.se <= true.value[6]) & (results.out$sigma22+t05*c
     .louis$sigma22.se >= true.value[6])))/NUM.DATA
c.rho.x  <- 100*(sum((results.out$get.rho-t05*c.louis$get.rho.se <= true.value[7]) & (results.out$get.rho+t05*c
     .louis$get.rho.se >= true.value[7])))/NUM.DATA
coverage <- c(c.beta0, c.beta1, c.eta, c.alpha, c.theta1,c.theta2, c.rho.x)
options(scipen=3)
final.out <- cbind(true.value, mean.out, median.out, sd.out, avg.se.out, percent.bias, mse.out,   coverage)
print(final.out, digits=3)
parameters         <- c("Beta0","Beta1","eta", "alpha","theta1","theta2","rho")
final.out2 <- cbind(true.value, mean.out, sd.out, avg.se.out, percent.bias, mse.out,   coverage)
final.out3 <- cbind(true.value, mean.out, sd.out, avg.se.out, percent.bias, mse.out )
sink(file=paste(directory,"G",n.cluster,"_n.obs",n.obs,"_final_summary_xtable_out.txt",sep=""))
  print(paste("G=",n.cluster," n=",n.obs," alpha=",T.alpha," results", sep=""))
 final.out
   print(paste("G=",n.cluster," n=",n.obs," alpha=",T.alpha," results", sep=""))
  print(xtable(final.out, digits=c(7,0,3,3,3,3,1,3,1)))#
 final.out3
   print(paste("G=",n.cluster," n=",n.obs," alpha=",T.alpha," results", sep=""))
   print(xtable(final.out3, digits=c(7,0,3,3,3,1,3)))#
 final.out2
   print(paste("G=",n.cluster," n=",n.obs," alpha=",T.alpha," results", sep=""))
   print(xtable(final.out2, digits=c(7,2,3,3,3,1,3,1)))#
sink()
################################################################################
# Plot the distribution of parameters for any G=  n=
################################################################################
 lwd.par <-1
 lty.par <-2
 par(mfrow=c(4,2))
 hist(results.out[,1], prob=TRUE, xlab=expression(paste("estimated ", beta,"0")), main=" ")
 mtext(paste("G=",n.cluster,"  n=",n.obs,"  alpha=", T.alpha, "  ARMS=",ARMsteps, "  EM steps=", EMsteps,sep=""
      ), side=3, line=1, outer=F, cex=0.9)
 lines(density(results.out[,1]), col="red", lty=2, lwd=lwd.par)
 curve(dnorm(x, mean=mean(results.out[,1]), sd=sd(results.out[,1])),  add=TRUE, col="blue", lwd=1)
 hist(results.out[,2], prob=TRUE, xlab=expression(paste("estimated ",beta,"1")), main=" " )
 lines(density(results.out[,2]), col="red", lty=2, lwd=lwd.par)
 curve(dnorm(x, mean=mean(results.out[,2]), sd=sd(results.out[,2])),  add=TRUE, col="blue")
 hist(results.out[,3], prob=TRUE, xlab=expression(paste("estimated ",eta)), main=" " )
 lines(density(results.out[,3]), col="red", lty=2, lwd=lwd.par)
 curve(dnorm(x, mean=mean(results.out[,3]), sd=sd(results.out[,3])),  add=TRUE, col="blue")
 hist(results.out[,4], prob=TRUE, xlab=expression(paste("estimated ",alpha)), main=" ")
 lines(density(results.out[,4]), col="red", lty=2, lwd=lwd.par)
 curve(dnorm(x, mean=mean(results.out[,4]), sd=sd(results.out[,4])),  add=TRUE, col="blue")
 hist(results.out[,5], prob=TRUE,  xlab=expression(paste("estimated ",theta,"1")), main=" ",   axes=TRUE)
 lines(density(results.out[,5] ), col="red", lty=2, lwd=lwd.par)
 curve(dnorm(x, mean=mean(results.out[,5]), sd=sd(results.out[,5])),  add=TRUE, col="blue")
 hist(results.out[,6], prob=TRUE, xlab=expression(paste("estimated ",theta,"2")), main=" ")
 lines(density(results.out[,6]), col="red", lty=2, lwd=lwd.par)
 curve(dnorm(x, mean=mean(results.out[,6]), sd=sd(results.out[,6])),  add=TRUE, col="blue")
 hist(results.out[,7], prob=TRUE, xlab=expression(paste("estimated ",rho)), main=" ")
 lines(density(results.out[,7]), col="red", lty=2, lwd=lwd.par)
 curve(dnorm(x, mean=mean(results.out[,7]), sd=sd(results.out[,7])),  add=TRUE, col="blue")
# use option paper="a4r" for landscape
pdf(file=paste(directory,"G",n.cluster,"_n.obs",n.obs,"_final_histogram.pdf",sep=""),onefile=TRUE ,paper="a4",
     height=11,width=8 )
  par(mfrow=c(4,2))
 hist(results.out[,1], prob=TRUE, xlab=expression(paste("estimated ", beta,"0")), main=" ")
  mtext(paste("G=",n.cluster,"  n=",n.obs,"  alpha=", T.alpha, "  ARMS=",ARMsteps, "  EM steps=", EMsteps,sep="
       "), side=3, line=1, outer=F, cex=0.9)
 lines(density(results.out[,1]), col="red", lty=2, lwd=lwd.par)
 curve(dnorm(x, mean=mean(results.out[,1]), sd=sd(results.out[,1])),  add=TRUE, col="blue", lwd=1)
 hist(results.out[,2], prob=TRUE, xlab=expression(paste("estimated ",beta,"1")), main=" " )
```

```
lines(density(results.out[,2]), col="red", lty=2, lwd=lwd.par)
curve(dnorm(x, mean=mean(results.out[,2]), sd=sd(results.out[,2])),  add=TRUE, col="blue")
hist(results.out[,3], prob=TRUE, xlab=expression(paste("estimated ",eta)), main=" " )
lines(density(results.out[,3]), col="red", lty=2, lwd=lwd.par)
curve(dnorm(x, mean=mean(results.out[,3]), sd=sd(results.out[,3])),  add=TRUE, col="blue")
hist(results.out[,4], prob=TRUE, xlab=expression(paste("estimated ",alpha)), main=" ")
lines(density(results.out[,4]), col="red", lty=2, lwd=lwd.par)
curve(dnorm(x, mean=mean(results.out[,4]), sd=sd(results.out[,4])),  add=TRUE, col="blue")
hist(results.out[,5], prob=TRUE,  xlab=expression(paste("estimated ",theta,"1")), main=" ",   axes=TRUE)
lines(density(results.out[,5] ), col="red", lty=2, lwd=lwd.par)
curve(dnorm(x, mean=mean(results.out[,5]), sd=sd(results.out[,5])),  add=TRUE, col="blue")
hist(results.out[,6], prob=TRUE, xlab=expression(paste("estimated ",theta,"2")), main=" ")
lines(density(results.out[,6]), col="red", lty=2, lwd=lwd.par)
curve(dnorm(x, mean=mean(results.out[,6]), sd=sd(results.out[,6])),  add=TRUE, col="blue")
hist(results.out[,7], prob=TRUE, xlab=expression(paste("estimated ",rho)), main=" ")
lines(density(results.out[,7]), col="red", lty=2, lwd=lwd.par)
curve(dnorm(x, mean=mean(results.out[,7]), sd=sd(results.out[,7])),  add=TRUE, col="blue")
dev.off()
```

./Louis_method_after_EM.R

BIBLIOGRAPHY

[1] O.O Aalen. *Statistical inference for a family of counting processes*. PhD thesis, University of California, Berkeley, 1975.

[2] H. Akaike. Information theory and an extension of the maximum likelihood principle. In *International Symposium on Information Theory, 2 nd, Tsahkadsor, Armenian SSR*, pages 267–281, 1973.

[3] J.E. Anderson and T.A. Louis. Survival analysis using a scale change random effects model. *Journal of the American Statistical Association*, pages 669–679, 1995.

[4] J.C. Biscarat, G. Celeux, and J. Diebolt. Stochastic versions of the EM algorithm. Technical report, University of Washington, Seattle, 1992.

[5] J. Buckley and I. James. Linear regression with censored data. *Biometrika*, 66(3):429–436, 1979.

[6] K.P. Burnham and D.R. Anderson. *Model selection and multi-model inference: a practical information-theoretic approach*. Springer, 2002.

[7] J.B. Carlin. *Seasonal analysis of economic time series, Unpublished Doctoral Dissertation*. PhD thesis, Department of Statistics, Stanford University, 1987.

[8] G. Celeux and J. Diebolt. The SEM algorithm: a probabilistic teacher algorithm derived from the EM algorithm for the mixture problem. *Computational Statistics Quarterly*, 2(1):73–82, 1985.

[9] D.G. Clayton. A Monte Carlo method for Bayesian inference in frailty models. *Biometrics*, 47(2):467–485, 1991.

[10] D.R. Cox. Regression models and life-tables. *Journal of the Royal Statistical Society. Series B (Methodological)*, 34(2):187–220, 1972.

[11] D.M. Dabrowska and K.A. Doksum. Estimation and testing in a two-sample generalized odds-rate model. *Journal of the American Statistical Association*, 83(403):744–749, 1988.

[12] A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 1–38, 1977.

[13] J. Diebolt and G. Celeux. Asymptotic properties of a stochastic EM algorithm for estimating mixing proportions. *Stochastic Models*, 9(4):599–613, 1993.

[14] J. Diebolt and EHS Ip. A stochastic EM algorithm for approximating the maximum likelihood estimate. Technical report, Department of Statistics, Standford University, Stanford, 1994.

[15] B. Efron. Missing data, imputation and the bootstrap. Technical report, Division of Biostatistics, Standford University, Stanford, 1992.

[16] B. Fisher, J. Costantino, C. Redmond, R. Poisson, D. Bowman, J. Couture, N.V. Dimitrov, N. Wolmark, D.L. Wickerham, E.R. Fisher, et al. A randomized clinical trial evaluating tamoxifen in the treatment of patients with node-negative breast cancer who have estrogen-receptor–positive tumors. *New England Journal of Medicine*, 320(8):479–484, 1989.

[17] B. Fisher, J. Dignam, J. Bryant, A. DeCillis, D.L. Wickerham, N. Wolmark, J. Costantino, C. Redmond, E.R. Fisher, D.M. Bowman, et al. Five versus more than five years of tamoxifen therapy for breast cancer patients with negative lymph nodes and estrogen receptor-positive tumors. *Journal of the National Cancer Institute*, 88(21):1529–1542, 1996.

[18] B. Fisher, J. Dignam, J. Bryant, and N. Wolmark. Five versus more than five years of tamoxifen for lymph node-negative breast cancer: updated findings from the national surgical adjuvant breast and bowel project b-14 randomized trial. *Journal of the National Cancer Institute*, 93(9):684–690, 2001.

[19] T.R. Fleming and D.Y. Lin. Survival analysis in clinical trials: past developments and future directions. *Biometrics*, 56(4):971–983, 2000.

[20] A. Gelman, J.B. Carlin, H.S. Stern, and D.B. Rubin. Bayesian Data Analysis. *Chapman&Hall/CRC*, 1995.

[21] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. *Journal of Applied Statistics*, 20(5):25–62, 1993.

[22] S.G. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–741, 1984.

[23] W.R. Gilks, N.G. Best, and K.K.C. Tan. Adaptive rejection Metropolis sampling within Gibbs sampling. *Applied Statistics*, 44(4):455–472, 1995.

[24] W.R. Gilks and P. Wild. Adaptive rejection sampling for Gibbs sampling. *Applied Statistics*, 41(2):337–348, 1992.

[25] I.D. Ha, Y. Lee, and J.K. Song. Hierarchical-likelihood approach for mixed linear models with censored data. *Lifetime data analysis*, 8(2):163–176, 2002.

[26] D.P. Harrington and T.R. Fleming. A class of rank test procedures for censored survival data. *Biometrika*, 69(3):553, 1982.

[27] P. Hougaard. A class of multivanate failure time distributions. *Biometrika*, 73(3):671–678, 1986.

[28] P. Hougaard. Survival models for heterogeneous populations derived from stable distributions. *Biometrika*, 73(2):387–396, 1986.

[29] P. Hougaard. Frailty models for survival data. *Lifetime data analysis*, 1(3):255–273, 1995.

[30] P. Hougaard. Fundamentals of survival data. *Biometrics*, 55(1):13–22, 1999.

[31] P. Hougaard. *Analysis of Multivariate Survival Data*. Springer New York, 2000.

[32] J.P Hughes. Mixed effects models with censored data with application to HIV RNA levels. *Biometrics*, 55(2):625–629, 1999.

[33] E.H.S. Ip. *A stochastic EM estimator in the presence of missing data: Theory and applications*. PhD thesis, Department of Statistics, Stanford University, 1994.

[34] J.H. Jeong, S.H. Jung, and S. Wieand. A parametric model for long-term follow-up data from phase III breast cancer clinical trials. *Statistics in medicine*, 22(3):339–352, 2003.

[35] E.L. Kaplan and P. Meier. Nonparametric estimation from incomplete observations, JASA 53 (1958). *Journal of the American Statistical Association*, pages 457–481, 1958.

[36] N. Keiding, P.K. Andersen, and J.P. Klein. The role of frailty models and accelerated failure time models in describing heterogeneity due to omitted covariates. *Statistics in Medicine*, 16(2):215–224, 1997.

[37] J.P. Klein. Semiparametric estimation of random effects using the Cox model based on the EM algorithm. *Biometrics*, pages 795–806, 1992.

[38] J.P. Klein, C. Pelz, and M. Zhang. Modeling random effects for censored data by a multivariate normal regression model. *Biometrics*, 55(2):497–506, 1999.

[39] A. Komárek and E. Lesaffre. Bayesian accelerated failure time model for correlated censored data with a normal mixture as an error distribution. *Statistica Sinica*, 17:549–569, 2007b.

[40] A. Komárek, E. Lesaffre, and J.F. Hilton. Accelerated failure time model for arbitrarily censored data with smoothed error distribution. *Journal of Computational and Graphical Statistics*, 14(3):726–745, 2005.

[41] A. Komárek, E. Lesaffre, and C. Legrand. Baseline and treatment effect heterogeneity for survival times between centers using a random effects accelerated failure time model with flexible error distribution. *Statistics in medicine*, 26(30):5457–5472, 2007a.

[42] A. Kottas and A.E. Gelfand. Bayesian semiparametric median regression modeling. *Journal of the American Statistical Association*, 96(456):1458–1468, 2001.

[43] N.M. Laird and J.H. Ware. Random-effects models for longitudinal data. *Biometrics*, pages 963–974, 1982.

[44] P. Lambert, D. Collett, A. Kimber, and R. Johnson. Parametric accelerated failure time models with random effects and an application to kidney transplant survival. *Statistics in medicine*, 23(20):3177–3192, 2004.

[45] K. Lange. A quasi-Newton acceleration of the EM algorithm. *Statistica sinica*, 5(1):1–18, 1995.

[46] K. Lange. Optimization,. *New York: Springer-Verlag*, 2004.

[47] E.W. Lee, LJ Wei, and Z. Ying. Linear regression analysis for highly stratified failure time data. *Journal of the American Statistical Association*, pages 557–565, 1993.

[48] Y. Lee and J.A. Nelder. Hierarchical generalized linear models. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 619–678, 1996.

[49] K.Y. Liang and S.L. Zeger. Longitudinal data analysis using generalized linear models. *Biometrika*, 73(1):13–22, 1986.

[50] T.A. Louis. Finding the observed information matrix when using the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 44(2):226–233, 1982.

[51] N. Mantel. Evaluation of survival data and two new rank order statistics arising in its consideration. *Cancer chemotherapy reports. Part 1*, 50(3):163, 1966.

[52] C.A. McGilchrist and C.W. Aisbett. Regression with frailty in survival analysis. *Biometrics*, pages 461–466, 1991.

[53] I. Meilijson. A fast improvement to the EM algorithm on its own terms. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 127–138, 1989.

[54] X.L. Meng and D.B. Rubin. Using EM to obtain asymptotic variance-covariance matrices: The SEM algorithm. *Journal of the American Statistical Association*, 86(416):899–909, 1991.

[55] S.F. Nielsen. The stochastic EM algorithm: estimation and asymptotic results. *Bernoulli*, 6(3):457–489, 2000.

[56] D. Oakes. Biometrika centenary: survival analysis. *Biometrika*, 88(1):99, 2001.

[57] W. Pan and J.E. Connett. A multiple imputation approach to linear regression with clustered censored data. *Lifetime data analysis*, 7(2):111–123, 2001.

[58] W. Pan and T.A. Louis. A Linear Mixed-Effects Model for Multivariate Censored Data. *Biometrics*, 56(1):160–166, 2000.

[59] AN Pettitt. Censored observations, repeated measures and mixed effects models: An approach using the EM algorithm and normal errors. *Biometrika*, 73(3):635–643, 1986.

[60] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2011. ISBN 3-900051-07-0.

[61] S.W. Raudenbush and A.S. Bryk. *Hierarchical linear models: Applications and data analysis methods*. Sage Publications, 2002.

[62] S.G. Self and K.Y. Liang. Asymptotic properties of maximum likelihood estimators and likelihood ratio tests under nonstandard conditions. *Journal of the American Statistical Association*, 82(398):605–610, 1987.

[63] P.J. Solomon. Effect of misspecification of regression models in the analysis of survival data. *Biometrika*, 71(2):291, 1984.

[64] D.O. Stram and J.W. Lee. Variance components testing in the longitudinal mixed effects model. *Biometrics*, pages 1171–1177, 1994.

[65] M.A. Tanner. *Tools for statistical inference: methods for the exploration of posterior distributions and likelihood functions*. Springer Verlag, 1996.

[66] M.A. Tanner and W.H. Wong. The calculation of posterior distributions by data augmentation. *Journal of the American statistical Association*, pages 528–540, 1987a.

[67] A.A. Tsiatis. Estimating regression parameters using linear rank tests for censored data. *The Annals of Statistics*, 18(1):354–372, 1990.

[68] F. Vaida and S. Blanchard. Conditional Akaike information for mixed-effects models. *Biometrika*, 92(2):351–370, 2005.

[69] F. Vaida and R. Xu. Proportional hazards model with random effects. *Statistics in Medicine*, 19(24):3309–3324, 2000.

[70] S. Walker and B.K. Mallick. A bayesian semiparametric accelerated failure time model. *Biometrics*, 55(2):477–483, 1999.

[71] G.C.G. Wei and M.A. Tanner. A Monte Carlo implementation of the EM algorithm and the poor man's data augmentation algorithms. *Journal of the American Statistical Association*, 85(411):699–704, 1990.

[72] L.J. Wei. The accelerated failure time model: a useful alternative to the cox regression model in survival analysis. *Statistics in Medicine*, 11(14-15):1871–1879, 1992.

[73] X. Xue and R. Brookmeyer. Bivariate frailty model for the analysis of multivariate survival time. *Lifetime Data Analysis*, 2(3):277–289, 1996.

[74] D. Zhang and X. Lin. Variance component testing in generalized linear mixed models for longitudinal/clustered data and other related topics. *Random effect and latent variable model selection*, pages 19–36, 2008.